A NUMERICAL INVESTIGATION OF FINITE-AMPLITUDE DISTURBANCES IN A PLANE POISEUILLE FLOW

by

George Donoghue O'Brien

United States
Naval Postgraduate School

THESIS

A NUMERICAL INVESTIGATION OF FINITE-AMPLITUDE DISTURBANCES IN A PLANE POISEUILLE FLOW

by

George Donoghue O'Brien

June 1970

This document has been approved for public release and sale; its distribution is unlimited.

T137213

A Numerical Investigation of Finite-Amplitude Disturbances in a Plane Poiseuille Flow

by

George Donoghue O'Brien
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1960

Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

from the
NAVAL POSTGRADUATE SCHOOL
June 1970

LIBRARY
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIF. 93940

ABSTRACT

A consistent and stable finite-difference approximation of the vorticity transport equations has been applied to a plane Poiseuille flow with spatially periodic disturbances of finite amplitude.

By application of the discrete Fourier transform to the solution of a Poisson equation significant improvements in the speed and accuracy of the calculations have been obtained. Numerical tests indicate that the usual iterative schemes are inadequate if a large number of mesh points are employed.

The effect of mesh size has been determined by extensive calculations based on a consistent second-order representation of the linear eigenvalue problem. The results indicate that a fine mesh is required to accurately represent the behavior of even the large-scale unstable motions. The stability and accuracy of the present formulation was demonstrated, in the linear range, by comparison with the established results of the linear theory.

Calculations in the non-linear range, at a moderate value of the Reynolds number, indicate the existence of an equilibrium state. The calculated values of the mean spectral density of kinetic energy are consistent with the exponential law predicted by Kraichnan for two-dimensional turbulence. The program, developed in this investigation, provides a means, of demonstrated accuracy, for the extensive investigation of two-dimensional turbulence in a shear flow.

TABLE OF CONTENTS

I. INTRODUCTION 11

A. BACKGROUND AND OBJECTIVES 11

B. DIFFERENCES BETWEEN TWO AND THREE-DIMENSIONAL UNSTEADY MOTION 14

C. PREVIOUS NUMERICAL WORK 15

1. Preliminary Remarks 15

2. Summary of Selected Numerical Investigations 17

II. STATEMENT OF THE PROBLEM AND BASIC EQUATIONS 21

III. ANALYSIS OF THE TRANSFORMED EQUATIONS 28

A. BASIC EQUATIONS IN TERMS OF FOURIER COMPONENTS 28

B. LINEAR THEORY 30

C. NON-LINEAR ANALYSIS 36

IV. THE FINITE-DIFFERENCE EQUATIONS AND SOLUTION 40

A. CHOICE OF THE METHOD 40

B. THE FINITE-DIFFERENCE EQUATIONS 45

C. CONVERGENCE OF THE NUMERICAL SOLUTION 53

1. Consistency 54

2. Stability Analysis of an Analogous Linear Equation 56

3. Stability and Accuracy of the Complete Non-Linear Equations 62

D. SOLUTION OF POISSONS' EQUATION FOR THE STREAM FUNCTION . . 67

E. DESCRIPTION OF THE PROGRAM AND COMPUTATIONAL PROCEDURE 77

F. FINITE-DIFFERENCE REPRESENTATION OF THE LINEAR EIGENVALUE PROBLEM 80

V. RESULTS AND DISCUSSION 87

A. LINEAR CALCULATIONS 87

B. NUMERICAL INTEGRATIONS 98

1. Preliminary 98

2. Results 100

VI. CONCLUSIONS 126

VII. RECOMMENDATIONS FOR FUTURE WORK 128

APPENDIX A - Derivation of Discrete Relations for the Spectral Densities of Squared Vorticity and Kinetic Energy . 129

APPENDIX B - Listing of the FORTRAN Code for the Numerical Integration 132

APPENDIX C - Listing of the FORTRAN Code for the Solution of the Algebraic Eigenvalue Problem 179

BIBLIOGRAPHY 200

INITIAL DISTRIBUTION LIST 204

FORM DD 1473 205

LIST OF TABLES

Table		Page
I	Computer Storage Requirements	42
II	Results of Test Solutions of a Poisson Equation	75
III	Coefficients for Vorticity Boundary Conditions	82
IV	Summary of Numerical Integrations	100

LIST OF FIGURES

Figure		Page
1	Curve of Neutral Stability for Plane Poiseuille Flow	34
2	Coordinates and Domain	46
3	Finite Difference Mesh	46
4	Effect of Mesh Size on Curve of Neutral Stability	90
5	Effect of N on Calculated Growth Rates	91
6	Calculated Eigenvalues, R = 6666.7, α = 1.0	92
7	Calculated Eigenvalues, R = 11333.3, α = 1.0	93
8	Calculated Eigenvalues, R = 25000, α = 1.0	94
9	Comparison of Calculated Growth Rates, R = 11333.3	95
10	Comparison of Calculated Growth Rates, R = 7333.3	96
11	Comparison of Calculated Eigenvectors, R = 6666.7, α = 1.0	97
12	Test for Linear Solution, Run 1	107
13	Test for Linear Solution, Run 2	108
14	Distortion of Eigenvector after 1000 time steps, Run 1	109
15	Effect of Amplitude on Initial Growth Rate	110
16	Energies vs. Time, Run 7	111
17a	Energies vs. Time, Run 8	112
17b	Energies vs. Time, Run 8	113
17c	Energies vs. Time, Run 8	114
18a	Energies vs. Time, Run 9	115
18b	Energies vs. Time, Run 9	116

19	Mean Velocity Profiles, Run 8	117
20	Mean Velocity, Vorticity, Turbulent Energy, Run 8	118
21	Spectral Energy Density, $y = \pm .85$, Run 8	119
22	Spectral Energy Density, Mean Values, Run 8	120
23	Initial Amplitude and Phase, Mode 1, Run 8	121
24	Final Amplitude and Phase, Mode 1, Run 8	122
25	Final Amplitude and Phase, Mode 2, Run 8	123
26	Final Amplitude and Phase, Mode 3, Run 8	124
27	Final Amplitude and Phase, Mode 4, Run 8	125

ACKNOWLEDGEMENTS

The author expresses his gratitude and appreciation to the following:

Dr. Theodore H. Gawain for his supervision and aid, and for giving so generously of his knowledge and time during the entire course of this work.

Drs. Daniel J. Collins, Frank D. Faulkner, James A. Miller and James V. Saunders for their service on the examination committee and for their useful criticisms.

Dr. Roger T. Williams for his encouragement, and a number of valuable suggestions.

The staff of the Computer Facility of the Naval Postgraduate School, and, in particular, the Operations Manager, Mr. David F. Norman, for their aid and cooperation in making available large amounts of computer time.

Finally to his wife, Elise, and his children for their encouragement and patient acceptance of the demands this undertaking put upon him.

I. INTRODUCTION

A. BACKGROUND AND OBJECTIVES

The phenomenon of turbulent flow has been the subject of numerous theoretical and experimental investigations since the publication of the results of the definitive experiments of Reynolds (1883). Yet, despite these efforts, our knowledge and ability to predict quantities of technical importance in a turbulent shear flow remain severely limited.

The fundamental difficulty encountered in theoretical investigations of turbulence arises from the non-linear character of the equations of motion. The operation of statistical averaging introduces an additional dependent variable, the Reynolds stress, to the equations for the mean flow. The dynamic equation for the Reynolds stress involves higher order correlations, and so on. Although the basic equations are determinate, the statistically averaged equations are not. This is the so-called closure problem of the statistical theory. Hypotheses for the closure of the averaged equations range from the mixing length approach of Prandtl (1925) to the rather formidable Direct-Interaction approximation of Kraichnan (1964).

The advent of the high-speed digital computer has excited interest in a new approach to the problem of turbulence, the numerical integration of the basic equations by means of a suitable finite-difference representation. The potential value of such an approach lies not in the determination of the mean quantities in a particular flow, but rather in the possibility of determining, subject to the uncertainties which beset any numerical experiment, detailed and comprehensive

information about the structure of the turbulent motion. This information would be useful for the development of suitable models for the effects of the turbulence, and for the testing of existing closure hypotheses.

The numerical approach is not without serious difficulties. The representation of the wide range of scales of motion involved in a turbulent flow produces staggering demands on the capacity and speed of the computer. In fact, the requirements are far beyond what is presently available (Orsag 1969). However, such calculations, at moderate values of the Reynolds number, may be possible within a decade.

A second difficulty concerns the development and testing of a finite-difference model which is stable, accurate and, in terms of computational requirements, efficient. A rigorous proof of the stability and convergence of finite-difference approximations to non-linear partial differential equations does not seem possible. Certain necessary conditions can be obtained by analysis of the linearized equations, but sufficient conditions are generally not available. The usual recourse is to demonstrate that the numerical solution does give a known result in some special case. A number of distinct finite-difference representations have been developed and employed for fluid dynamics problems, each with different strong points and weaknesses. There does not seem to be any general agreement as to which approach might be the best for a given application.

The present capabilities of digital computers do permit numerical integrations for three-dimensional shear flows if a fairly coarse mesh is employed. In a number of recent investigations attempts have been

made to determine the behavior of the mean flow by means of a finite-difference representation which includes a model for the subgrid-scale motion (Deardorff 1970), (Galloway 1968). These are considered in somewhat greater detail in Section C.2. There are three fundamental difficulties associated with such undertakings. The first is the present lack of a sufficiently accurate model for the subgrid-scale motion. Secondly, the effect of the mesh size on the model for the subgrid-scale motion, and on the mean flow is largely undetermined. Finally the requirements for the accurate representation of even large-scale motions in the high shear layers near the boundaries may be greater than the coarse mesh permits. Despite these difficulties, qualitatively satisfying results have been obtained.

Another approach, which is within the capabilities of current digital computers, is the numerical integration of the two-dimensional equations. There are fundamental and significant differences between two- and three-dimensional unsteady motions, which are considered briefly in Section B. However, the equally fundamental non-linearity of the equations remains. Two-dimensional disturbances in a parallel shear flow will, to a limited degree, excite motions of different scale and create apparent stresses which deform the mean flow. The results of a numerical simulation of a two-dimensional turbulent motion might be useful, to some degree, for the development of a suitable model for the effects of the small scales of motion, and the testing of various hypotheses concerning the structure of the turbulence.

Considerations affecting the finite-difference formulation and solution are qualitatively the same for the two-dimensional problem as for the ultimate three-dimensional one. Whatever difficulties beset

the former will be increased in the latter, due to the greatly increased number of mesh points. Schemes, which are unsuitable in the two-dimensional case, will almost certainly be unsuitable in the three-dimensional one. Schemes which are stable and accurate in two dimensions are likely to be stable and accurate in three, although the fairly well-known stability constraints will be more severe in the three-dimensional case.

For the case of a channel flow with periodic boundary conditions upstream and downstream, the two-dimensional problem offers the possibility of testing the stability and accuracy of the linear features of the formulation by comparison with the established results of the Orr-Sommerfeld theory.

The present investigation involves just such a numerical integration for a two-dimensional plane Poiseuille flow with periodic disturbances. The objectives are:

1. To develop a stable, accurate and efficient finite-difference formulation for the plane Poiseuille flow, and to test its accuracy in a long-term integration by comparison with established results from the linear theory.
2. To apply the method for the determination of finite-amplitude effects on the stability of small disturbances.
3. To determine, if possible, an equilibrium condition for the two-dimensional flow by long-term integration with a growing disturbance.

B. DIFFERENCES BETWEEN TWO- AND THREE-DIMENSIONAL UNSTEADY MOTION

There are fundamental differences between the kinds of non-linear interactions that can take place in two- and three-dimensional unsteady

flows. The effects of these differences are often described by noting the absence of "vortex stretching" in a two-dimensional flow. For zero viscosity, the circulation about all closed curves which move with a fluid particle remains constant. In a two-dimensional flow, all such curves remain in the plane, and the well-known result is the invariance of the vorticity of a particle. A three-dimensional flow is capable of bending or twisting a closed curve which moves with a particle. The curl of the velocity vector can change its direction: the component of vorticity in a specified plane can change.

Kraichnan (1967) has shown that as a result of the existence of local inviscid constants, a two-dimensional flow admits of two formal inertial ranges, the usual -5/3 range and an additional -3 range. The -3 range, in this case, involves a cascade of vorticity to higher wave numbers, and the -5/3 range a cascade of energy to lower wave numbers, larger scales of motion.

C. PREVIOUS NUMERICAL WORK

1. Preliminary Remarks

The equations of motion for an incompressible flow can be written in the familiar way, in terms of the velocity vector and the pressure.

$$\frac{\partial \bar{V}}{\partial t} + (\bar{V}\cdot\nabla)\,\bar{V} = -\nabla P + \nu\nabla^2\bar{V} \tag{1.1}$$

with the additional requirement that

$$\nabla\cdot\bar{V} = 0 \tag{1.2}$$

to satisfy the conservation of mass. The equation for the pressure is found by taking the divergence of equation (1.1).

$$\nabla\cdot[(\bar{V}\cdot\nabla)\bar{V}] = -\nabla^2 P \tag{1.3}$$

Another formulation, which implicitly satisfies the conservation of mass, is in terms of the curl of the velocity vector, $\overline{\Gamma}$, and a vector velocity potential, $\overline{\psi}$, which may be regarded as a generalized stream function.

$$\frac{\partial \overline{\Gamma}}{\partial t} = - \left[(\nabla x \overline{\psi}) \cdot \nabla \right] \overline{\Gamma} + (\Gamma \cdot \nabla) \ (\nabla x \overline{\psi}) + \nu \nabla^2 \Gamma \tag{1.4}$$

where

$$\overline{\Gamma} = \nabla (\nabla \cdot \overline{\psi}) - \nabla^2 \overline{\psi} \tag{1.5}$$

and

$$\overline{V} = \nabla x \overline{\psi} \tag{1.6}$$

Without restricting the solution, $\overline{\psi}$ can be specified to be non-divergent (Aziz 1967). Then (1.5) becomes

$$\overline{\Gamma} = - \nabla^2 \overline{\psi} \tag{1.7}$$

In the case of a two-dimensional motion, this formulation becomes the ordinary stream function-vorticity expression.

Finite-difference approximations to the equations of motion in terms of $\overline{V}$ and P generally involve a staggered mesh arrangement in which pressures are defined at one family of mesh points and velocities at the other. The treatment of the boundary conditions for the pressure at the walls requires the establishment of virtual mesh points outside the domain and somewhat arbitrary specifications for the values of velocity at those points. Conservation of mass is not satisfied exactly, but the solution of a Poisson equation (1.3) for the pressure is designed so that at each point the divergence of the velocity is given a rate of change that will tend to correct for the accumulated divergence at that point.

Finite-difference representations based on equations (1.4) and (1.7) conserve mass implicitly. It is generally possible to derive a consistent explicit treatment of the boundary values for the vorticity without defining virtual points outside of no-slip boundaries. For three-dimensional integrations based on these equations a corrective procedure, similar to the one employed for the pressure calculation above, is required to maintain the zero divergence of $\overline{\psi}$.

The essential mathematical features of both approaches are the same. At each time step a forecast for new values of $\overline{V}$ or $\overline{\Gamma}$ is obtained from the finite-difference representation of (1.1) or (1.4). The solution of a Poisson equation for the pressure in the first case, or three such equations for the components of $\overline{\psi}$ in the second, completes a time step. It is the Poisson solution which generally takes up most of the computing time.

2. Summary of Selected Numerical Investigations

In a very well-known work, Fromm (1963) has employed a finite-difference representation based on the vorticity transport equations to follow the development of the vortex street behind a rectangular obstacle in a two-dimensional flow. The agreement between his computed streak lines and photographs of similar real flows is striking. The finite-difference representation involved an energy-conserving representation of the non-linear terms, an explicit formulation of boundary values of the vorticity at the moving walls, and periodic boundary conditions upstream and downstream. The time differencing was by the Du Fort-Frankel method. The domain of numerical stability was determined by analysis of the linearized equations and extensive numerical experiment. The finite-difference mesh contained 25 x 57 points. The

Poisson equation for the stream function was solved by Gauss-Seidel iteration.

Perhaps the best-known finite-difference representation based on the momentum equations is the Marker and Cell method of Harlow and Welsh (1965). The scheme, which was designed to be suitable for the representation of a flow with a free surface, employs a staggered mesh in which the components of velocity and the pressure are defined at distinct families of points. The treatment of solid boundaries requires the establishment of virtual points outside the domain, and somewhat arbitrary relations between values at the virtual and interior points. The failure of the representation of the non-linear terms to conserve kinetic energy is an important defect for reasons which will be considered in Chapter IV. Reported calculations employed first-order forward time differencing. The Poisson equation for the pressure was solved by Gauss-Seidel iteration.

An investigation which is closely related to the present work is due to Dixon (1966). It is an application of finite-difference methods to the problem of spatially-growing disturbances in Poiseuille and plane Poiseuille flow. The finite-difference equations are based on a version of the vorticity transport equations with an explicit separation of the laminar components. The representation of the non-linear terms failed to conserve kinetic energy. In addition the particular first-order explicit formulation of the boundary values of vorticity was inconsistent with the otherwise second-order finite-difference representation. Alternating direction methods were employed for the time differencing and the Poisson equation was solved by Successive Over-relaxation. For the plane Poiseuille flow Dixon assumed symmetry about the midstream, and employed a mesh with external

streamwise dimension 37 times the radius. The mesh contained 10 points between the centerline and the wall, and 296 in the direction of the stream. Conditions at the entrance region were specified, and the evolution of the disturbance was followed downstream. The results were somewhat uncertain. No evidence of growing disturbances was found. At very low values of the Reynolds number the disturbance decayed. In all other cases the behavior of the disturbance was characterized by periodic amplification and decay.

Pearson (1964) has considered several finite difference approaches to the solution of a test problem. He asserted that if the representation is based on the vorticity transport equations, an iterative method for the time step, which includes a method of forecasting new values of the vorticity at the boundaries, is required. This was based on the observation that the basic equations do not include a specification of boundary values of the vorticity. It turned out, however, that the calculations were unstable unless the forecast was "smoothed" by weighting it with a fraction of the old value, and that the required weighting factor was in excess of 80 per cent. The procedure did produce a more rapid convergence of the iterative solution to the implicit equations for the time step. However, the results do not seem to constitute a convincing argument for the necessity of an iterative treatment of vorticity at no-slip boundaries.

Galloway (1968) has employed a second-order finite-difference representation of the momentum equations in an application to the three-dimensional flow in a channel of square cross section. The approach employs Du Fort-Frankel time differencing and an energy-conserving

representation of the non-linear terms. The Poisson equation for the pressure was solved by an interesting iterative procedure which takes advantage of the weak coupling between various groups of the algebraic equations. Calculations were performed in meshes containing 9^3, 11^3 and 17^3 points. It was noted that the resolution was inadequate to accurately represent the unsteady motion, however refinements of the mesh produced marked improvements in the results. An integral representation of the sub-grid-scale motion was proposed for subsequent calculations.

Very recently Deardorff (1970) has performed calculations for a similar channel flow, at a high Reynolds number, which employed such an integral representation of the sub-grid-scale motion. The results are qualitatively satisfying, although it was noted that a number of improvements in the sub-grid model are needed.

II. STATEMENT OF THE PROBLEM AND BASIC EQUATIONS

The equations of motion for a two-dimensional incompressible flow are

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} = - \frac{\partial P}{\partial x} + \frac{1}{R} \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \tag{2.1}$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} = - \frac{\partial P}{\partial y} + \frac{1}{R} \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right) \tag{2.2}$$

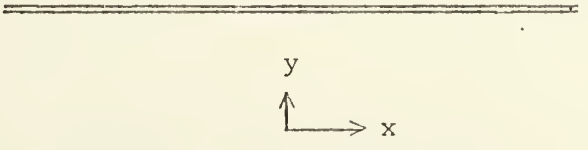
where u and v are the dimensionless components of velocity, P is the pressure and R is the Reynolds number which is defined by

$$R = \frac{VL}{\nu} \tag{2.3}$$

where V is the unit of velocity, L is the unit of length and ν is the kinematic viscosity. The equation of continuity is

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{2.4}$$

We consider the two-dimensional flow between fixed parallel planes. The origin of coordinates is midway between the planes with the x-axis parallel and the y-axis normal to them.



Let the reference unit of length be one-half the distance between the parallel planes, and the reference unit of velocity be the mean value of the steady laminar flow velocity distribution.

Then the laminar flow velocity distribution and pressure gradient are given by

$$U(y) = -\frac{3}{2}(y^2 - 1) \tag{2.5}$$

$$\frac{dP}{dx} = \frac{3}{R} \tag{2.6}$$

If for the general case we let

$$u(x,y,t) = U(y) + u'(x,y,t) \tag{2.7}$$

$$v(x,y,t) = v'(x,y,t) \tag{2.8}$$

$$p(xyt) = P(x) + p'(xyt) \tag{2.9}$$

Then equations (2.1), (2.2) and (2.3) become

$$\frac{\partial u'}{\partial t} + U\frac{\partial u'}{\partial x} + v'\frac{dU}{dx} + u'\frac{\partial u'}{\partial x} + v'\frac{\partial u'}{\partial y} = -\frac{\partial P}{\partial x} - \frac{\partial p'}{\partial x} + \frac{1}{R}\left(\frac{\partial^2 u'}{\partial x^2} + \frac{\partial^2 u'}{\partial y^2}\right) \tag{2.10}$$

$$\frac{\partial v'}{\partial t} + U\frac{\partial v'}{\partial x} + u'\frac{\partial v'}{\partial x} + v'\frac{\partial v'}{\partial y} = -\frac{\partial P}{\partial y} - \frac{\partial p'}{\partial y} + \frac{1}{R}\left(\frac{\partial^2 v'}{\partial x^2} + \frac{\partial^2 v'}{\partial y^2}\right) \tag{2.11}$$

$$\frac{\partial u'}{\partial x} + \frac{\partial v'}{\partial y} = 0 \tag{2.12}$$

The no-slip boundary conditions require that at $y = \pm 1$

$$u' = v' = 0$$

An equation for the disturbance pressure can be obtained by differentiating equation (2.10) with respect to x, equation (2.11) with respect to y, and summing. Thus

$$\frac{\partial^2 p'}{\partial x^2} + \frac{\partial^2 p'}{\partial y^2} = 2\left[\frac{\partial u'}{\partial x}\frac{\partial v'}{\partial y} - \left(\frac{dU}{dy} + \frac{\partial u'}{\partial y}\right)\frac{\partial v'}{\partial x}\right] \tag{2.13}$$

We can, without confusion, delete the primes in the subsequent notation for the disturbance quantities.

If we specify in addition that the disturbance makes no contribution to the total mass flow;

i.e. that $$\int_{-1}^{1} u(x,y,t)\, dy = 0 \tag{2.14}$$

then the reference unit of velocity, as previously defined, is as well the mean transport velocity. This conforms to the usual choice of reference velocity for turbulent channel flows.

A more compact, and for purposes of computing, more efficient formulation of the basic equations is obtained by introducing the stream function and the vorticity.

Let $$u = -\frac{\partial \psi}{\partial y},\ v = \frac{\partial \psi}{\partial x} \tag{2.15}$$

$$\Gamma = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \tag{2.16}$$

Then equation (2.12) is implicitly satisfied. Equations (2.10) and (2.11) become

$$\frac{\partial \Gamma}{\partial t} + U \frac{\partial \Gamma}{\partial x} - U'' \frac{\partial \psi}{\partial x} - \frac{\partial \psi}{\partial y}\frac{\partial \Gamma}{\partial x} + \frac{\partial \psi}{\partial x}\frac{\partial \Gamma}{\partial y} = \frac{1}{R}\left(\frac{\partial^2 \Gamma}{\partial x^2} + \frac{\partial^2 \Gamma}{\partial y^2}\right) \tag{2.17}$$

where

$$\Gamma = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} \tag{2.18}$$

and

$$U'' = \frac{d^2 U}{dy^2} = -3$$

The no-slip boundary conditions require that at $y = \pm 1$

$$\frac{\partial \psi}{\partial x} = \frac{\partial \psi}{\partial y} = 0 \tag{2.19}$$

Equation (2.14) becomes

$$\int_{-1}^{1} \frac{\partial \psi}{\partial y}\, dy = \psi(x,+1,t) - \psi(x,-1,t) = 0 \tag{2.20}$$

Thus the no-slip boundary conditions become

$$\psi(x,+1,t) = \psi(x,-1,t) = \text{const} \tag{2.21}$$

$$\frac{\partial \psi}{\partial y}(x,\pm 1,t) = 0. \tag{2.22}$$

If we consider a finite region of the space between the parallel planes, and specify the initial state of the disturbance and the conditions at the upstream and downstream boundaries of the region, then, in principle, the evolution of the disturbance can be determined by means of a suitable numerical method. Unfortunately the question of the proper boundary conditions upstream and downstream is somewhat moot. A reasonable recourse is to assume that the motion is periodic in the direction of the mean flow. Thus we consider a disturbance such that

$$\psi(x \pm 2nh,y,t) = \psi(x,y,t) \tag{2.23}$$

$$\Gamma(x \pm 2nh,y,t) = \Gamma(x,y,t) \tag{2.24}$$

$$n = 0, 1, 2, \cdots$$

where the basic period is 2h.

It should be emphasized that, in the present formulation, the disturbance quantities do not describe a two-dimensional turbulent motion, but rather the total departure of the motion from the steady laminar flow. The turbulent motion is described by the difference between the disturbance and an appropriately defined mean value of the disturbance.

In the present case, as a consequence of the assumed periodicity of the motion, the disturbance is homogeneous in the streamwise direction. That is, all statistical correlations are independent of the origin of the x coordinate. Than an appropriate definition of the statistical mean value of a disturbance quantity is the average value over a period in the x direction.

Some remarks about notation are in order. It has become necessary to distinguish between quantities associated with the laminar flow, the disturbance, the mean flow and the turbulence. Some indulgence on the part of the reader will be required.

With the exception of certain quadradic quantities such as turbulent energy or Reynolds stress, mean values, obtained by means of the linear operator

$$\frac{1}{2h}\int_{-h}^{h} dx$$

are denoted by an overbar. All quantities associated with the two-dimensional turbulence are denoted by a circumflex.

The distinction between local and mean values of quadradic quantities associated with the turbulence is indicated in the arguments of the various functions, or in the case of finite-difference expressions, the subscripts. Thus, since the averaging operator is linear,

$$\begin{aligned}
\hat{u}\,(x,y,t) &= u\,(x,y,t) - \overline{u}\,(y,t)\\
\hat{v}\,(x,y,t) &= v\,(x,y,t) - \overline{v}\,(y,t)\\
\hat{\psi}\,(x,y,t) &= \psi\,(x,y,t) - \overline{\psi}\,(y,t)\\
\hat{\Gamma}\,(x,y,t) &= \Gamma\,(x,y,t) - \overline{\Gamma}\,(y,t)
\end{aligned} \tag{2.25}$$

where

$$\begin{aligned}
\hat{u}\,(x,y,t) &= -\frac{\partial}{\partial y}\hat{\psi}\,(x,y,t)\\
\overline{u}\,(y,t\ \ ) &= -\frac{\partial}{\partial y}\overline{\psi}\,(y,t\ \ )\\
\hat{v}\,(x,y,t) &= \ \ \frac{\partial}{\partial x}\hat{\psi}\,(x,y,t)\\
\overline{v}\,(y,t\ \ ) &= \ \ \frac{\partial}{\partial x}\overline{\psi}\,(y,t\ \ ) \equiv 0
\end{aligned} \tag{2.26}$$

The mean flow velocity and vorticity are given by

$$w\ (y,t) = U\ (y) - \frac{\partial}{\partial y}\ \overline{\psi}\ (y,t) \tag{2.27}$$

$$\Omega\ (y,t) = -\ U'' + \overline{\Gamma}\ (y,t) \tag{2.28}$$

It is convenient at this point to define a number of quantities which are calculated in the numerical scheme, and to derive expressions for them in terms of the stream function. The kinetic energy in the mean flow is

$$\begin{aligned} E\ (y,t) &= 1/2\ w^2\ (y,t) \\ &= 1/2\left[U(y) - \frac{\partial}{\partial y}\ \overline{\psi}\ (y,t)\right]^2 \end{aligned} \tag{2.29}$$

The normalized sum of mean flow kinetic energy over a domain which includes one period of the disturbance is

$$\begin{aligned} E(t) &= \frac{1}{2h} \int_{-h}^{h} dx\ \frac{1}{2} \int_{-1}^{1} E(y,t)\ dy \\ &= \frac{1}{2} \int_{-1}^{1} E(y,t)\ dy \end{aligned} \tag{2.30}$$

The kinetic energy in the turbulent motion is

$$\hat{E}\ (x,y,t) = \frac{1}{2}\left[\left(-\frac{\partial\hat{\psi}}{\partial y}\right)^2 + \left(\frac{\partial\hat{\psi}}{\partial x}\right)^2\right] \tag{2.31}$$

The mean value is

$$\hat{E}\ (y,t) = \frac{1}{2h} \int_{-h}^{h} \hat{E}\ (x,y,t)\ dx \tag{2.32}$$

And the normalized sum of turbulent kinetic energy over the domain is

$$\hat{E}(t) = \frac{1}{2h} \int_{-h}^{h} dx \, \frac{1}{2} \int_{-1}^{1} \hat{E}(x,y,t)\, dy$$

$$= \frac{1}{2} \int_{-1}^{1} \hat{E}(y,t)\, dy \qquad (2.33)$$

Finally the Reynolds stress produced by the two-dimensional turbulent motion is

$$S(y,t) = \frac{1}{2h} \int_{-h}^{h} \frac{\partial \hat{\psi}}{\partial x} \frac{\partial \hat{\psi}}{\partial y}(x,y,t)\, dx$$

III. ANALYSIS OF THE TRANSFORMED EQUATIONS

A. BASIC EQUATIONS IN TERMS OF FOURIER COMPONENTS

The equations of motion for the plane Poiseuille flow with periodic disturbances can also be expressed in terms of the wave number in the x direction by means of the finite Fourier Transform.

Let

$$f_n(y,t) = \frac{1}{2h} \int_{-h}^{h} \psi\ (x,y,t)\ e^{i\alpha nx}\ dx$$

$$g_n(y,t) = \frac{1}{2h} \int_{-h}^{h} \Gamma\ (x,y,t)\ e^{i\alpha nx}\ dx \tag{3.1}$$

Then

$$\psi\ (x,y,t) = \sum_{n=-\infty}^{\infty} f_n(y,t)\ e^{-i\alpha nx}$$

$$\Gamma\ (x,y,t) = \sum_{n=-\infty}^{\infty} g_n(y,t)\ e^{-i\alpha nx} \tag{3.2}$$

where $\alpha = \pi/h$. Since both ψ and Γ are real,

$$f_{-n}(y,t) = f_n^*\ (y,t)$$

$$g_{-n}(y,t) = g_n^*\ (y,t) \tag{3.3}$$

where the asterisk indicates a complex conjugate.

By means of the transform operator

$$\frac{1}{2h} \int_{-h}^{h} e^{i\alpha nx}\ dx$$

and the associated orthogonality relation

$$\frac{1}{2h}\int_{-h}^{h} e^{-i\alpha(n-s)x}\,dx = 1 \quad \text{if } s = n$$

$$= 0 \quad \text{otherwise}$$

it can be shown that equations (2.17) and (2.18) are satisfied if

$$(g_n'' - \alpha^2 n^2 g_n) - R\frac{\partial}{\partial t} g_n + i\alpha nR\left[(U-f_o')\, g_n - (U''-f_o''')\, f_n\right]$$

$$- i\alpha R\, H_n = 0 \tag{3.4}$$

$$n = 0,\ 1,\ 2,\ \cdots$$

where

$$g_n = f_n'' - \alpha^2 n^2 f_n \tag{3.5}$$

$$H_n = \sum_{s=1}^{\infty}\left\{(n-s)\,(f_s'\, g_{n-s} - g_s'\, f_{n-s}) + (n+s)\,(f_s^{*\prime}\, g_{n+s} - g_s^{*\prime} f_{n+s})\right\} \tag{3.6}$$

and primes denote derivatives with respect y. The no-slip boundary conditions require that at $y = \pm 1$

$$f_n = f_n' = 0 \tag{3.7}$$

$$n = 0,\ 1,\ \cdots$$

The functions $f_o(y,t)$ and $g_o(y,t)$ represent the mean values of the disturbance stream function and vorticity respectively. In terms of these variables, the mean velocity, equation (2.27), is given by

$$w(y,t) = U(y) - f_o'(y,t) \tag{3.8}$$

The non-linear interaction between the nth mode of the disturbance with the distorted mean flow is represented by the term

$$(U-f_o')\, g_n - (U''-f_o''')\, f_n$$

in equation (3.4). The non-linear interaction between the nth mode and the other disturbance modes is given by H_n. The equation for the

mean flow is, from equations (3.4), (3.5) and (3.6),

$$f_o''' - R\frac{\partial}{\partial t} f_o'' - i\alpha R \sum_{s=1}^{\infty} \left\{ s \left[(g_s f_s^*) - (g_s f_s^*)^* \right]' \right\} = 0 \qquad (3.9)$$

Equations (3.4) and (3.7) govern the behavior of periodic disturbances of finite amplitude in a plane Roiseuille flow. They are an infinite set of coupled non-linear partial differential equations.

Previous analysis of the finite-amplitude disturbances has generally involved the truncation of this set of equations, the deletion of certain non-linear terms to uncouple the remaining equations and various assumptions about the time dependence of the remaining modes. A brief description of the most prominent investigations is given in Section C.

B. LINEAR THEORY

If the amplitude of the disturbance is sufficiently small so that the terms quadradic in the amplitude, the non-linear terms, may be ignored in comparison with the linear terms, then equations (3.4) and (3.5) become

$$(g_n'' - \alpha^2 n^2 g_n) - R\frac{\partial}{\partial t} g_n + i\alpha n R \left[U g_n - U'' f_n \right] = 0 \qquad (3.10)$$

$$g_n = f_n'' - \alpha^2 n^2 f_n \qquad (3.11)$$

In this case the solutions for the various modes are uncoupled. By absorbing the index n into the definition of α, and assuming the separation of variables

$$\begin{aligned} f(y,t) &= \Phi(y)\, e^{i\beta t} \\ g(y,t) &= \theta(y)\, e^{i\beta t} \end{aligned} \qquad (3.12)$$

where β is, in general, complex.

Equations (3.10) and (3.11) become

$$(\theta'' - \alpha^2\theta) + i\alpha R\left[(U-\beta/\alpha)\ \theta - U''\Phi\right] = 0 \tag{3.13}$$

$$\theta = \Phi'' - \alpha^2\ \Phi \tag{3.14}$$

Or in terms of Φ alone,

$$(\Phi^{IV} - 2\alpha^2\Phi'' + \alpha^4\Phi) + i\alpha R\left[(U-\beta/\alpha)\ (\Phi''-\alpha^2\Phi) - U''\Phi\right] = 0 \tag{3.15}$$

The boundary conditions, equation (3.7), require that at $y = \pm 1$

$$\Phi = \Phi' = 0 \tag{3.16}$$

If the spatial wave number, α, and the Reynolds number, R, are specified, then equation (3.15) has solutions which satisfy the homogeneous boundary conditions only for discrete values of the complex number β. Equation (3.15) is the well-known Orr-Sommerfeld equation.

From equation (3.15) it is evident that, if the imaginary part of β is negative, the amplitude of the disturbance increases with time. The disturbance is said to be unstable. If the imaginary part is positive the disturbance is stable. Of particular interest are conditions of neutral stability. The largest value of the Reynolds number at which, for all α, all the eigenvalues of equation (3.15) are stable is termed the critical Reynolds number. Schensted (1960) has shown that for a plane Poiseuille flow the eigenfunctions of equation (3.15) form a complete set. Any continuous function which satisfies the boundary conditions, equations (3.16), can be represented as a linear combination of them. Thus if, at given values of α and R, all of the eigensolutions of equation (3.15) are stable, the flow is stable with respect to all infinitesmal disturbances.

The solutions of equation (3.15) for the case of plane Poiseuille flow have been extensively investigated. The zone of linear instability in the Reynolds number-wave number domain, and the associated curve of neutral stability, have been established in a number of investigations, the most prominent of which is due to Lin (1945). Following a procedure based on asymptotic expansions, which was outlined by Heisenberg (1924), Lin calculated values of α and R on the neutral curve and determined a critical Reynolds number (in terms of the reference length and velocity of the present investigation) of 3540 at a wave number of 1.04.

Thomas (1953) employed a basically fourth-order finite-difference representation of equation (3.15) for the determination of the primary eigenvalues associated with the symmetric modes at eighteen points in the vicinity of the critical point. The discrete representation of the eigenfunction involved 101 points between the centerline and the wall. The resulting algebraic equations were solved by direct Gauss elimination. The calculations, which were performed on an early electronic calculator, required two weeks of machine time. He determined by interpolation a critical value of the Reynolds number equal to 3850 at a wave number of 1.04. For the calculation at $R = 6666.7$ and $\alpha = 1.0$ the values of the primary eigenfunction at the 101 grid points are tabulated.

Potter (1966) has used methods generally similar to those of Lin, but with a different expansion for the inviscid solution, to investigate the linear modes of a combined plane Couette-Poiseuille flow. His results for the Poiseuille flow indicate that there are slight errors in Lin's neutral curve for $\alpha > .8$. Potter suggested that this was due to the slow convergence of Lin's expansion in that region.

More recently Grosch (1968) has employed an expansion of the solutions of equation (3.15) in terms of the known eigenfunctions of

$$\Phi''' - \alpha^2 \Phi'' + \alpha^4 \Phi = \lambda^4 \Phi$$

$$\Phi (\pm 1) = \Phi' (\pm 1) = 0$$

The expansion was truncated generally after thirty terms, and the resulting algebraic eigenvalue problem was solved by means of the Q R algorithm (Wilkinson 1965). His results showed excellent agreement with those of Thomas for both the various eigenvalues and the tabulated eigenvector at $R = 6666.7$, $\alpha = 1$. The results for the neutral curve confirm Potter's observations. Solutions are obtained for both symmetric and antisymmetric modes. Instabilities are found only for the symmetric modes, and in no case is more than one of the calculated modes, at a given α and R, unstable.

A portion of the neutral curve for the Plane Poiseuille flow based on the data of Lin and Grosch is shown in Fig. 1.

All reference to Reynolds numbers and dimensionless wave numbers has been in terms of the reference length and velocity employed in this investigation, one half the channel width and the mean laminar velocity. In the investigations of Lin and Thomas the reference unit of velocity is the maximum laminar velocity. Based on that value, the Reynolds number is exactly 1.5 times values given here. In the works of Grosch and Potter (for the case of zero Couette component) the reference unit of length is the channel width, and the reference velocity is the maximum laminar velocity. Wave numbers are twice those given here, and Reynolds numbers three times those given here.

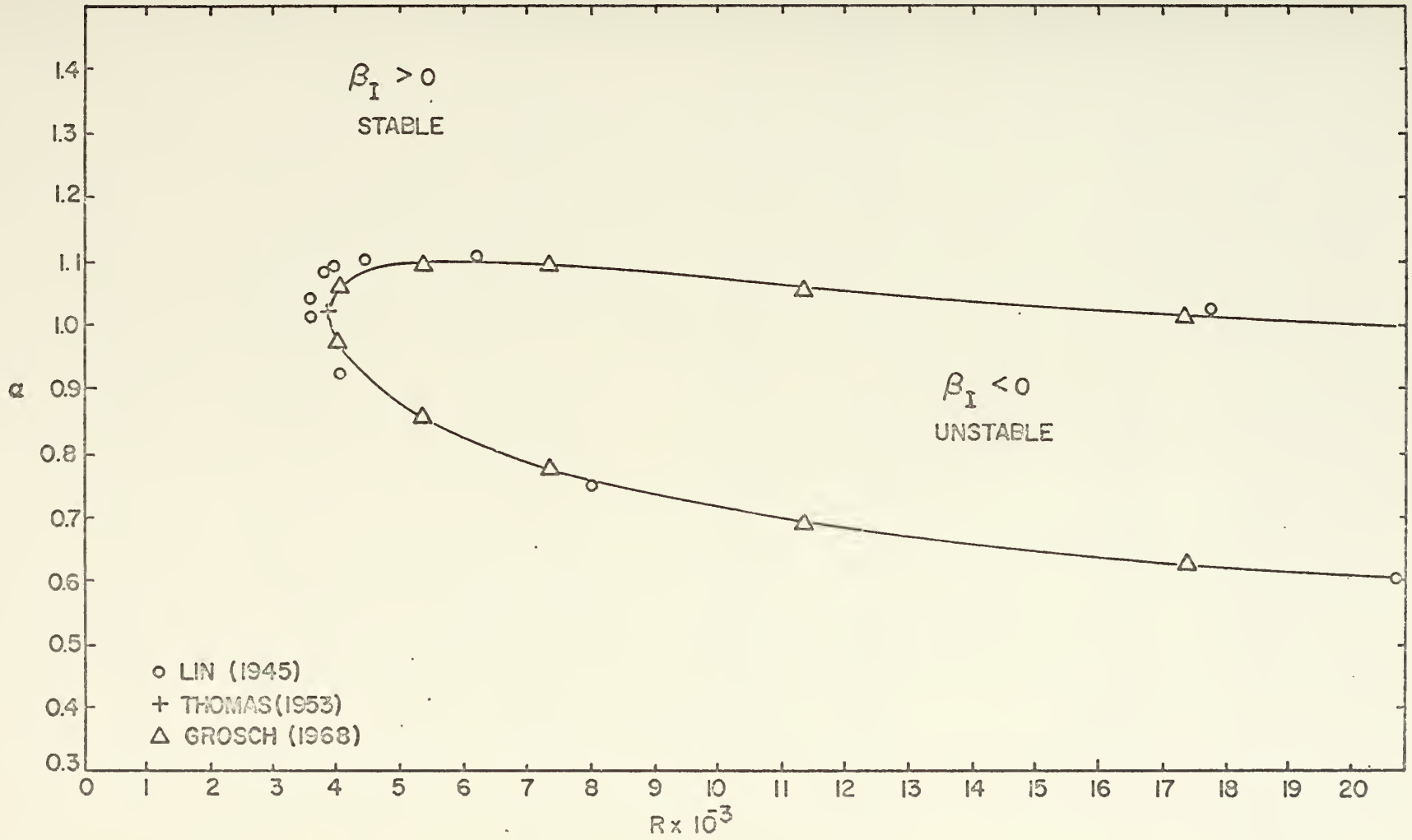


FIGURE 1 CURVE OF NEUTRAL STABILITY FOR PLANE POISEUILLE FLOW.

It is convenient, and useful for the interpretation of some of the results of the numerical integrations of the linearized equations, to indicate at this point certain relations for the total kinetic energy in the linear solutions. Equations (3.2) and (3.12) for the linear solutions are equivalent to

$$\psi(x,y,t) = 2e^{-\beta_I t} \left[\Phi_R(y) \cos(\alpha x - \beta_R t) + \Phi_I(y) \sin(\alpha x - \beta_R t) \right] \qquad (3.17)$$

where

$$\beta = \beta_R + i\beta_I$$

$$\Phi = \Phi_R \& i\Phi_I$$

It is easy to show that, if $h = \pi/\alpha$, the total disturbance energy, equation (2.33) is given by

$$\hat{E}(t) = \alpha/2 \; e^{-2\beta_I t} \int_{-1}^{1} \left[(\Phi_R'^2 + \Phi_I'^2) + \alpha^2 (\Phi_R^2 + \Phi_I^2) \right] dy \qquad (3.18)$$

$$= \text{const} \times e^{-2\beta_I t}$$

The disturbance energy grows, or decays exponentially at twice the rate of the disturbance amplitude. The disturbance itself grows or decays exponentially and changes its phase at the rate β_R/α.

If, in the linear problem, the disturbance is composed of a number of the linear eigenfunctions with the same wave number, α, then the total disturbance energy will contain terms like equation (3.18) peculiar to each eigenfunction Φ and eigenvalue β, plus cross-coupling terms which involve pairs of the modes.

For example, if the disturbance is composed of two eigenfunctions

$$\psi(x,y,t) = \psi_1(x,y,t) + \psi_2(x,y,t)$$

where

$\Phi(y)$ and β describe ψ_1

$\eta(y)$ and γ describe ψ_2

Then it can be shown that the total disturbance energy is given by

$$\hat{E}(t) = \hat{E}_1(t) + \hat{E}_2(t) + \alpha\, e^{-(\beta_I+\gamma_I)t}\left\{A \cos\left[(\beta_R-\gamma_R)t\right] - B \sin\left[(\beta_R-\gamma_R)t\right]\right\}. \tag{3.19}$$

where $E_1(t)$ and $E_2(t)$ are from equation (3.18), and A and B are given by

$$A = \int_{-1}^{1}\left[(\Phi_R' \eta_R' + \Phi_I' \eta_I') + \alpha^2 (\Phi_R \eta_R + \Phi_I \eta_I)\right] dy$$

$$B = \int_{-1}^{1}\left[\Phi_R' \eta_I' - \Phi_I' \eta_R') - \alpha^2 (\eta_R \Phi_I - \Phi_R \eta_I)\right] dy.$$

Thus, in this case, the total disturbance energy contains terms with exponential growth rates $-2\beta_I$, $-2\gamma_I$ and $-(\beta_I + \gamma_I)$, and the last term oscillates with frequency $(\beta_R - \gamma_R)$.

If

$$-2\beta_I > -2\gamma_I$$

then eventually $\hat{E}_1(t)$ will overshadow the other terms.

C. NON-LINEAR ANALYSIS

A fundamental consequence of the non-linearity of the equations of motion is the production of the Reynolds stress and the resulting deformation of the mean flow. The distortion of the mean flow, which is dependent on the amplitude of the disturbance, can alter the rate of transfer of energy from the mean flow to the disturbance and, as a result, change its rate of growth or decay.

A second non-linear effect is the excitation of disturbance motions of other wave lengths. Dynamical constraints on the non-linear interactions in a two-dimensional motion limit the systematic transfer of kinetic energy to smaller scales of motion (Lilly 1965, Kraichnan 1967). As a result, it is expected that, in terms of the kinetic energy in the various disturbance modes, the primary mode will remain dominant. Thus the question of non-linear stability centers on the behavior of the amplitude of the primary mode. If, with increasing time, the amplitude approaches zero, it is stable; if not, it is unstable. Stuart (1958) has employed the terms subcritical and supercritical to describe the stable and unstable regions in the wave number-Reynolds number domain for infinitesmal (linear) disturbances. Subcritical instability refers to the growth of a disturbance of finite amplitude at a wave number and Reynolds number for which it is stable in the linear analysis.

A question separate from that of stability concerns the existence of an equilibrium state. It seems reasonable to expect that at some sufficiently large amplitude the non-linear interactions will limit the growth of the disturbance motion.

In an early investigation, Meksyn and Stuart (1958) employed a variation of the asymptotic methods of the linear problem which included the distortion of the mean velocity. Their results suggest the existence of subcritical instabilities at values of the Reynolds number as low as 1933.3.

Subsequently Stuart (1958) derived an approximate expression for the time dependence of the amplitude of the primary mode of the disturbance. The analysis is based on an energy balance and, as Stuart

indicates, may have questionable applicability to flows, such as the plane Poiseuille flow, for which the precise details of the mean motion are of crucial importance to the question of stability. In the development only the interaction between the primary mode and the mean flow is considered; the generation of higher harmonics in the disturbance motion is ignored. In addition it is assumed that the time derivative of the mean velocity is always negligible and that the phase velocity and the shape of the primary mode are as given by the linear theory.

With reference to equations (3.4) this is equivalent to

$$f_n(y,t) = 0 \qquad n = 2, 3, \cdots$$

$$f_1(y,t) = A(t)\, \Phi_1(y)\, e^{i\beta_R t}$$

$$\beta_R << 1$$

where $A(t)$ is an amplitude factor and $\Phi(y)$ and β_R are from the linear analysis. From the energy balance it is deduced that $A(t)$ satisfies an equation of the form

$$\frac{d(A^2)}{dt} = (K_1 - K_2)\, A^2 - K_3 A^4$$

where K_1 is associated with the energy production terms, K_2 with the viscous dissipation, and K_3 is determined by the non-linear interactions. At sufficiently small amplitudes the latter term is negligible and the sign of (K_1-K_2) determines the stability. If $K_3 < 0$, then regardless of the sign of (K_1-K_2) there is the possibility of finite amplitude instability.

In subsequent undertaking, Stuart (1960) and Watson (1960) derived a formal expansion for the solutions to the non-linear equations (3.4). Based on certain order-of-magnitude arguments, the Fourier expansion

(3.2) for the solution is truncated after n=2, and enough of the non-linear terms are deleted so that the remaining equations are uncoupled. A separation of variables is assumed for the functions $f_1(y,t)$ and $f_2(y,t)$, in which they are expressed in terms of perturbations of the corresponding linear solutions, and a complex amplitude function, A(t), which includes the changes in the phase speed due to the non-linear interactions. This function is assumed to satisfy

$$\frac{d|A|^2}{dt} = -\beta_I|A|^2 + a_1|A|^4 + \cdots \tag{3.20}$$

The result $a_1 < 0$ is taken to indicate either the existence of super-critical equilibrium condition or finite-amplitude stability in the subcritical domain. If $a_1 > 0$ then, in the subcritical case an unstable equilibrium is implied, and in the supercritical case this result is taken to indicate the absence of an equilibrium condition (at least at small amplitudes). The last conclusion must be qualified since if the disturbance continues to grow then eventually higher order terms in (3.20) will become significant: the assumptions upon which the formal expansion is based will no longer apply.

Reynolds and Potter (1967) have developed a somewhat more straightforward formulation of the Stuart-Watson expansion and performed the indicated calculations for the case of a combined plane Couette-Poiseuille flow. The results for Poiseuille flow indicate values of $a_1 < 0$ only near the lower branch of the neutral curve.

In similar but more extensive calculations, Pekeris and Shkoller (1967) found the zone of negative a_1 in the wave number-Reynolds number domain. It includes the lower branch, but excludes the vicinity of the upper branch of the curve of neutral stability for the linear problem.

IV. THE FINITE DIFFERENCE EQUATIONS AND SOLUTION

A. CHOICE OF THE METHOD

The formulation of the finite-difference model for the equations of motion is determined by considerations of accuracy, numerical stability and computational efficiency. The latter consideration refers to some weighted measure of computing time and storage requirement.

The fundamental equations of motion contain three dependent variables: pressure and two components of velocity. The vorticity transport equations contain two: stream function and vorticity. The essential mathematical features of each are the same: non-linear parabolic equations for the time derivatives; a linear elliptic equation for the pressure in the first case, and the stream function in the second. Thus, although there are important differences in the treatment of the boundary conditions, similar finite difference methods can be applied to each set of equations. The same kinds of stability and accuracy constraints on the finite-difference parameters, δx and δt, will occur in each case.

The stability and efficiency of the finite-difference representation of such equations is principally determined by the method of time differencing. A number of direct and implicit methods are available.

Consider equations of the form

$$\frac{\partial f}{\partial t} = D(f,g); \; A(f) = B(g)$$

where

$$f = f(x,y,t) \quad g = g(x,y,t)$$

and A, B and D are differential operators in x and y. For the discrete representation in t let

$$f(x,y,t_K) = f^K(x,y)$$

$$g(x,y,t_K) = g^K(x,y)$$

where

$$t_K = K\delta t$$

Both Forward, and Modified Euler implicit time differencing require the simultaneous storage of two levels of the variables involved in the time differencing.

$$\frac{f^{N+1} - f^N}{\delta t} = D\ (f^N g^N) + 0\ (\delta t)$$

$$\frac{f^{N+1} - f^N}{\delta t} = \frac{1}{2} D\ (f^{N+1} g^{N+1}) + \frac{1}{2} D\ (f^N g^N) + 0\ (\delta t^2)$$

If, in the case of Modified Euler differencing, the implicit equations for f^{N+1} are solved by Gauss-Seidel iteration then it can be shown that the total storage requirement is the same as for forward differencing.

Central time differencing requires the simultaneous storage of three levels of the variables involved in the time differencing.

$$\frac{f^{N+1} - f^{N-1}}{2\delta t} = D\ (f^N g^N) + 0\ (\delta t^2)$$

Thus, for an M x N finite-difference grid, the basic storage requirements may be summarized as follows.

TABLE I

Computer Storage Requirements

	Momentum Equations	Vorticity Transport Equations
Forward, or Modified Euler Time Differencing	5 (M x N)	3 (M x N)
Central Time Differencing	7 (M x N)	4 (M x N)

Clearly, in terms of the storage requirement, a formulation based on the vorticity transport equations is superior. In addition there are advantages in computing time. The velocity components, when they are required, can be obtained from the stream function with fewer operations than are required to perform the additional evaluation of a time derivative. A determination of the pressure field requires the additional solution of a Poisson equation. If this is required then some of these advantages disappear.

Of the three methods of time differencing considered, the Modified Euler method is, in principle, the most accurate and, in particular for non-linear problems such as this one, the least susceptible to numerical instability. However it is implicit: the solution of a set of simultaneous equations is required at each time step. The

most practical means of solving the equations is by Gauss Seidel Iteration or Successive Over-relaxation. Each step in the iteration will require a solution of the Poisson equation for the stream function; this demands all the operations that are involved in the course of a complete time step by an explicit method. Thus if an explicit method, of similar truncation error, can be made stable by reducing the time step by a factor less than the average number of iterations for the implicit case, it is to be preferred.

Forward time differencing is subject to linear stability constraints on the ratios $\delta t/\delta y$ and $\delta t/\delta y^2$, which are associated respectively with the diffusion and linear advection terms. Early experiments with this method indicated that even if these constraints were observed numerical instability occurred when the non-linear terms became sufficiently large.

If the ordinary second order finite difference representation of the diffusion term is employed, then the central time differencing is unstable at all values of δt. This instability is removed by the Du Fort-Frankel (1953) approximation to the diffusion term. There still remains a constraint on the ratio $\delta t/\delta y$, which is associated with the advection terms. Experiments were conducted with both the Central and the Modified Euler methods. At values of δt small enough so that the Central differencing is stable, the Modified Euler method required from six to eight iterations: that is between six and eight times the computational effort to produce essentially the same results. Increasing the magnitude of the time step produced a predictable instability with the Central differencing and an increased number of iterations for convergence with the Modified Euler method. At the

stability boundary for the Central differencing the ratio $\delta t/\delta y$ is of the order unity. Thus the Modified Euler method offers advantages only in a domain where the truncation errors associated with δt may be sufficiently large so as to obscure some transients in the solution. Accordingly, a formulation based on the vorticity transport equations, which employs the Du Fort-Frankel Central time differencing was used in this undertaking.

It should be noted that an important group of methods of time differencing has not been considered in the preceeding remarks. These are the Alternating Direction Implicit methods of Peaceman and Rachford (1955) and Douglas and Rachford (1956) and their variants. In each case the time increment is performed in two steps. In the first step, an intermediate value of the dependent variable is obtained by expressing certain of the spatial derivatives implicitly as in the Modified Euler method, and the others explicitly. In the second step the arrangement is reversed. Each step requires the solution of a set of simultaneous equations. The essential feature of the method is that these equations, in matrix form, are tri-diagonal: they may be solved directly and without storage penalty by a particularly simple case of Gauss Elimination (Richtmyer 1967). In applications to the equations for incompressible flow, the variations of the methods are due primarily to the particular treatment of the advection terms.

In the present investigation, the formulation of the upstream and downstream boundary conditions as well as the conservative finite difference representation of the advection terms introduce additional non-zero diagonals to the matrices which define the equations for each step. The matrices are no longer tri-diagonal: the essential advantages of the alternating direction methods do not obtain.

B. THE FINITE DIFFERENCE EQUATIONS

The computational domain, coordinates and the arrangement of the finite difference mesh are shown schematically in Figures (2) and (3). The mesh spacings are denoted by δx and δy, the value of the time step by δt. The mesh contains M x N points: M points in the x direction and N points in the y direction. The external dimensions are 2h and 2 respectively.

The particular treatment of the periodic boundary conditions requires the definition of virtual points just beyond the upstream and downstream boundaries. The boundaries lie midway between these points and the adjacent interior columns of points. The mesh is arranged so that points lie on the no-slip boundaries. Thus there are M intervals in the x direction and N-1 in the y direction.

Let

$$\Gamma^{\ell}_{K,J} = \Gamma\ (x_K, y_J, t_\ell)$$

$$\psi^{\ell}_{K,J} = \psi\ (x_K, y_J, t_\ell)$$

$$U_J = U(y_J)$$

where

$$\begin{aligned} x_K &= (K-1)\ \delta x - h + \delta x/2 \qquad K = 1,2,\cdots M \\ y_J &= (J-1)\ \delta y - 1 \qquad\qquad\ J = 1,2,\cdots N \end{aligned} \tag{4.1}$$

$$t_\ell = \sum_{n=1}^{\ell} \delta t_n$$

and

$$\begin{aligned} \delta x &= 2h/M \\ \delta y &= 2/(N-1) \end{aligned} \tag{4.2}$$

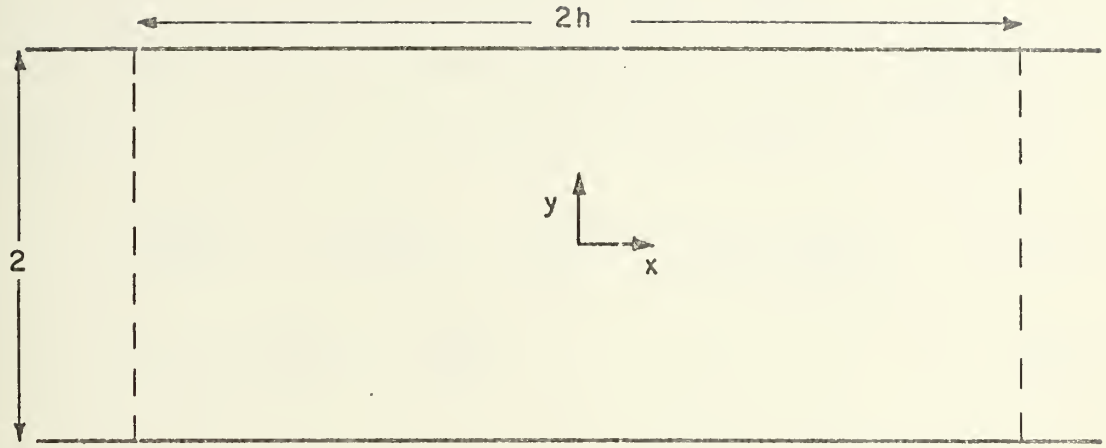


FIGURE 2 COORDINATES AND DOMAIN

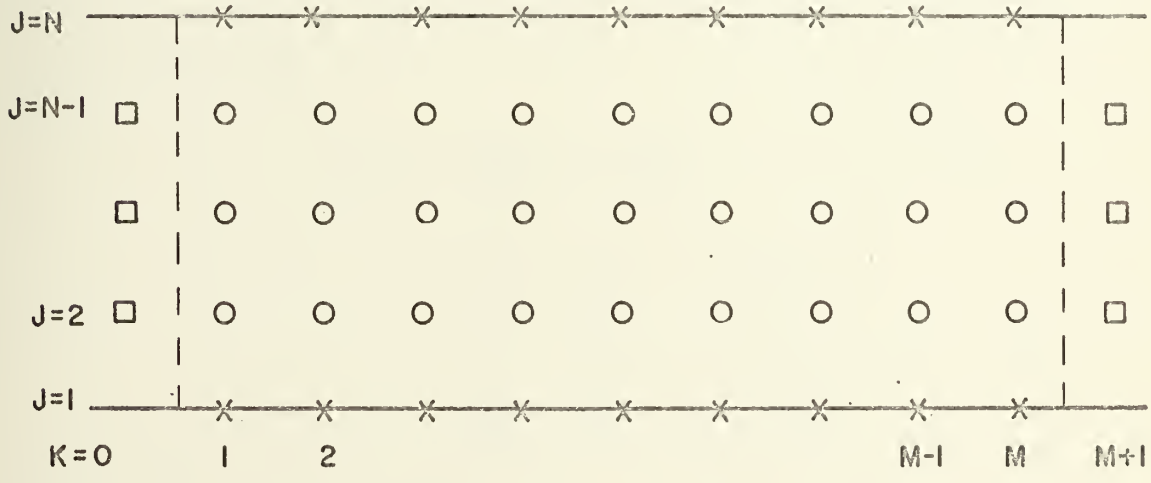


FIGURE 3 FINITE DIFFERENCE MESH

In the calculations M is always even, and N odd. Thus the origin of the coordinates is at the mesh point $K = M/2$, $J = \frac{N+1}{2}$.

The finite difference approximation to the vorticity transport equation is

$$\frac{\delta\Gamma^{t}}{\delta t}K,J = - U_J \frac{\delta\Gamma^{t}}{\delta x}K,J + U'' \frac{\delta\psi^{t}}{\delta x}K,J + J^{t}_{K,J} + \frac{1}{R} D^{t}_{K,J} \tag{4.3}$$

$$\Gamma^{t}_{K,J} = \frac{\delta^2\psi^{t}}{\delta x^2}K,J + \frac{\delta^2\psi^{t}}{\delta y^2}K,J \tag{4.4}$$

where $\frac{\delta}{\delta x}$, $\frac{\delta^2}{\delta x^2}$, etc., are the usual second-order central difference representations, $D^{t}_{K,J}$ is the Du Fort-Frankel approximation for the diffusion term and $J^{t}_{K,J}$ is the conservative representation due to Arakawa (1966). It may be regarded as a central difference approximation to the mean of three distinct, but equivalent, expressions for the non-linear terms.

$$\begin{aligned} J(\psi,\Gamma) &= \frac{\partial\psi}{\partial y}\frac{\partial\Gamma}{\partial x} - \frac{\partial\psi}{\partial x}\frac{\partial\Gamma}{\partial y} \\ &= \frac{\partial}{\partial y}\left(\psi \frac{\partial\Gamma}{\partial x}\right) - \frac{\partial}{\partial x}\left(\psi \frac{\partial\Gamma}{\partial y}\right) \\ &= \frac{\partial}{\partial x}\left(\Gamma \frac{\partial\psi}{\partial y}\right) - \frac{\partial}{\partial y}\left(\Gamma \frac{\partial\psi}{\partial x}\right) . \end{aligned} \tag{4.5}$$

In terms of the central difference operators

$$J^{t}_{K,J} = \frac{1}{3}\left(JA^{t}_{K,J} + JB^{t}_{K,J} + JC^{t}_{K,J}\right)$$

where

$$JA^{t}_{K,J} = \frac{\delta\psi^{t}}{\delta y}_{K,J} \frac{\delta\Gamma^{t}}{\delta x}_{K,J} - \frac{\delta\psi^{t}}{\delta x}_{K,J} \frac{\delta\Gamma^{t}}{\delta y}_{K,J}$$

$$JB^{t}_{K,J} = \frac{\delta}{\delta y}\left(\psi \frac{\delta\Gamma}{\delta x}\right)^{t}_{K,J} - \frac{\delta}{\delta x}\left(\psi \frac{\delta\Gamma}{\delta y}\right)^{t}_{K,J} \qquad (4.6)$$

$$JC^{t}_{K,J} = \frac{\delta}{\delta x}\left(\Gamma \frac{\delta\psi}{\delta y}\right)^{t}_{K,J} - \frac{\delta}{\delta y}\left(\Gamma \frac{\delta\psi}{\delta x}\right)^{t}_{K,J}$$

Each of the finite-difference forms, JA, JB and JC is a second order representation of the non-linear terms. The mean value has a number of important and desirable characteristics which are discussed in Section C.3.

In detail equations (4.3), (4.4) and (4.6).

$$\frac{\Gamma^{t+1}_{K,J} - \Gamma^{t-1}_{K,J}}{2\delta t} = - U_J \frac{\Gamma^{t}_{K+1,J} - \Gamma^{t}_{K-1,J}}{2\delta x} + U'' \frac{\psi^{t}_{K+1,J} - \psi^{t}_{K-1,J}}{2\delta x} + J^{t}_{K,J}$$

$$+ \frac{1}{R}\left[\frac{\Gamma^{t}_{K+1,J} - \left(\Gamma^{t+1}_{K,J} + \Gamma^{t-1}_{K,J}\right) + \Gamma^{t}_{K-1,J}}{\delta x^2}\right. \qquad (4.7)$$

$$\left. + \frac{\Gamma^{t}_{K,J+1} - \left(\Gamma^{t+1}_{K,J} + \Gamma^{t-1}_{K,J}\right) + \Gamma^{t}_{K,J-1}}{\delta y^2}\right] .$$

$$\Gamma^{t}_{K,J} = \frac{\psi^{t}_{K+1,J} - 2\psi^{t}_{K,J} + \psi^{t}_{K-1,J}}{\delta x^2} + \frac{\psi^{t}_{K,J+1} - 2\psi^{t}_{K,J} + \psi^{t}_{K,J-1}}{\delta y^2} \qquad (4.8)$$

$$J^t_{K,J} = \frac{1}{3}\left\{\left[\frac{\psi^t_{K,J+1} - \psi^t_{KmJ-1}}{2\delta x}\,\frac{\Gamma^t_{K+1,J} - \Gamma^t_{K-1,J}}{2\delta y} - \frac{\psi^t_{K+1,J} - \psi^t_{K-1,J}}{2\delta x}\,\frac{\Gamma^t_{K,J+1} - \Gamma^t_{K,J-1}}{2\delta y}\right]\right.$$

$$+\left[\left(\frac{\psi^t_{K,J+1}}{2\delta y}\,\frac{\Gamma^t_{K+1,J+1} - \Gamma^t_{K-1,J+1}}{2\delta x} - \frac{\psi^t_{K,J-1}}{2\delta y}\,\frac{\Gamma^t_{K+1,J-1} - \Gamma^t_{K-1,J-1}}{2\delta x}\right)\right.$$

$$\left.-\left(\frac{\psi^t_{K+1,J}}{2\delta x}\,\frac{\Gamma^t_{K+1,J+1} - \Gamma^t_{K+1,J-1}}{2\delta y} - \frac{\psi^t_{K-1,J}}{2\delta x}\,\frac{\Gamma^t_{K-1,J+1} - \Gamma^t_{K-1,J-1}}{2\delta y}\right)\right] \qquad (4.9)$$

$$+\left[\left(\frac{\Gamma^t_{K+1,J}}{2\delta x}\,\frac{\psi^t_{K+1,J+1} - \psi^t_{K+1,J-1}}{2\delta y} - \frac{\Gamma^t_{K-1,J}}{2\delta x}\,\frac{\psi^t_{K-1,J+1} - \psi^t_{K-1,J-1}}{2\delta y}\right)\right.$$

$$\left.\left.-\left(\frac{\Gamma^t_{K,J+1}}{2\delta y}\,\frac{\psi^t_{K+1,J+1} - \psi^t_{K-1,J+1}}{2\delta x} - \frac{\Gamma^t_{K,J-1}}{2\delta y}\,\frac{\psi^t_{K+1,J-1} - \psi^t_{K-1,J-1}}{2\delta x}\right)\right]\right\}$$

The periodic boundary conditions are approximated for equation (4.7) by assigning values of Γ and ψ at the virtual points as follows.

$$\psi^t_{0,J} = \psi^t_{M,J} \qquad \psi^t_{M+1,J} = \psi^t_{1,J} \qquad J = 1,2,\cdots N$$

$$\Gamma^t_{0,J} = \Gamma^t_{M,J} \qquad \Gamma^t_{M+1,J} = \Gamma^t_{M,J} \qquad J = 1,2,\cdots N \qquad (4.10)$$

Equation (3.8) is solved for $\psi^t_{K,J}$ by means of a discrete Fourier transform. The periodicity of Γ and ψ in the direction of the mean flow is implicit in the solution of that equation.

The no slip boundary conditions, equations (2.21) and (2.22), require that

$$\psi^t_{K,1} = \psi^t_{K,N} = \text{const} \qquad K = 1,2,\cdots M$$

and that, at the walls, the normal derivative of the stream function be zero. The first condition is satisfied most conveniently by assigning a zero value to the stream function at the wall. The second is employed

to derive a boundary condition for the vorticity in equation (4.7). Since, at the wall, all derivatives of the stream function with respect to x are zero, the vorticity there is given by

$$\Gamma_w = \frac{\partial^2 \psi}{\partial y^2} w.$$

By means of Taylor series expansions, the interior values of the stream function can be expressed in terms of the stream function and its derivatives at the wall. Based on these expansions, finite-difference expressions for Γ_w which satisfy

$$\frac{\partial \psi}{\partial y} w = 0 \tag{4.11}$$

can be derived. For example, at the lower wall (J=1), we have

$$\psi_2 = \psi_1 + \delta y \frac{\partial \psi}{\partial y} w + \frac{\delta y^2}{2} \frac{\partial^2 \psi}{\partial y^2} w + \frac{\delta y^3}{6} \frac{\partial^3 \psi}{\partial y^3} w + \frac{\delta y^4}{24} \frac{\partial^4 \psi}{\partial y^4} + \cdots$$

$$\psi_3 = \psi_1 + 2\delta y \frac{\partial \psi}{\partial y} w + 4 \frac{\delta y^2}{2} \frac{\partial^2 \psi}{\partial y^2} w + 8 \frac{\delta y^3}{6} \frac{\partial^3 \psi}{\partial y^3} w + 16 \frac{\delta y^4}{24} \frac{\partial^4 \psi}{\partial y^4} + \cdots$$

where, for simplicity, only the index J has been retained in the notation. Applying (4.11), we have, from the first expansion

$$\Gamma_w = \frac{\partial^2 \psi}{\partial y^2} w = \frac{2(\psi_2 - \psi_1)}{\delta y^2} + \frac{\delta y}{3} \left(\frac{\partial^3 \psi}{\partial y^3} w + \cdots \right. \tag{4.12}$$

Equation (4.12) is the first-order expression for the boundary values of the vorticity. It can also be obtained by applying equation (4.8) at the wall with the assumption of a virtual point outside the domain to satisfy (4.11). Although equation (4.8) is, in the interior region, of second order, the application at the boundary is not. A second order formula can be obtained by combining the two expansions.

$$\Gamma_w = \frac{8\,\psi_2 - \psi_3 - 7\,\psi_1}{2\delta y^2} + \frac{\delta y^2}{6}\left(\frac{\partial^4 \psi}{\partial y^4}\right) w + \cdots \qquad (4.13)$$

By including additional interior points third and fourth order formulas may be derived. They are

$$\Gamma_w = \frac{108\,\psi_2 - 27\,\psi_3 + 4\,\psi_4 - 85\,\psi_1}{18\,\delta y^2} + \frac{\delta y^3}{10}\left(\frac{\partial^5 \psi}{\partial y^5}\right) w + \cdots \qquad (4.14)$$

$$\Gamma_w = \frac{576\,\psi_2 - 216\,\psi_3 + 64\,\psi_4 - 9\,\psi_5 - 415\,\psi_1}{72\,\delta y^2} + \frac{\delta y^4}{15}\left(\frac{\partial^6 \psi}{\partial y^6}\right) + \cdots \qquad (4.15)$$

An integration based on equation (4.7) is not self-starting: another method is required for the first step. In addition the suppression of a certain weak instability associated with the central differencing requires the periodic averaging over two consecutive time steps and the performance of a step by another method. These points are considered in greater detail in Section C.3. The Modified Euler scheme is employed for these purposes. The associated finite difference equation is

$$\begin{aligned}
\frac{\Gamma^{t+1}_{K,J} - \Gamma^{t}_{K,J}}{\delta t} = \frac{1}{2}\Bigg\{ & - U_J \frac{\Gamma^{t+1}_{K+1,J} - \Gamma^{t+1}_{K-1,J}}{2\delta x} + U'' \frac{\psi^{t+1}_{K+1,J} - \psi^{t+1}_{K-1,J}}{2\delta x} + J^{t+1}_{K,J} \\
& + \frac{1}{R}\left[\frac{\Gamma^{t+1}_{k+1,J} - 2\Gamma^{t+1}_{K,J} + \Gamma^{t+1}_{K-1,J}}{\delta x^2} + \frac{\Gamma^{t+1}_{K,J+1} - \Gamma^{t+1}_{K,J} + \Gamma^{t+1}_{K,J-1}}{\delta y^2}\right]\Bigg\} \\
& + \frac{1}{2}\Bigg\{ - U_J \frac{\Gamma^{t}_{K+1,J} - \Gamma^{t}_{K-1,J}}{2\delta x} + U'' \frac{\psi^{t}_{K+1,J} - \psi^{t}_{K-1,J}}{2\delta x} + J^{t}_{K,J} \\
& + \frac{1}{R}\left[\frac{\Gamma^{t}_{K+1,J} - 2\Gamma^{t}_{K,J} + \Gamma^{t}_{K-1,J}}{\delta x^2} + \frac{\Gamma^{t}_{K,J+1} - 2\Gamma^{t}_{K,J} + \Gamma^{t}_{K,J-1}}{\delta y^2}\right]\Bigg\}
\end{aligned} \qquad (4.16)$$

where $J_{K,J}$ is as given by equation (4.9).

Finite-difference expressions for the various quantities which describe the mean flow and the two-dimensional turbulent motion follow in a straightforward way from equations (2.25) through (2.34).

In terms of the discrete representation the relation for the mean value of the disturbance stream function

$$\bar{\psi}\ (y,t) = \frac{1}{2h}\int_{-h}^{h} \psi\ (x,y,t)\ dx$$

becomes

$$\begin{aligned} \bar{\psi}^{\ell}_{J} &= \frac{1}{2h} \sum_{K=1}^{M} \psi^{\ell}_{K,J}\ \delta x \\ &= \frac{1}{M} \sum_{K=1}^{M} \psi^{\ell}_{K,J} \end{aligned} \tag{4.17}$$

Thus, in the discrete representation, the averaging operator simply implies the arithmetic mean value over the index K, or the zero mode in a discrete Fourier representation with respect to x (See Section D). The stream function which describes the turbulent motion is

$$\hat{\psi}^{\ell}_{K,J} = \psi^{\ell}_{K,J} - \bar{\psi}^{\ell}_{J} \tag{4.18}$$

The velocity and vorticity of the mean flow are given by

$$w^{\ell}_{J} = U_J - \frac{\bar{\psi}^{\ell}_{J+1} - \bar{\psi}^{\ell}_{J-1}}{2\delta y} \tag{4.19}$$

$$\Omega^{\ell}_{J} = - U'' + \bar{\Gamma}^{\ell}_{J} \tag{4.20}$$

Equation (2.29), for the mean flow kinetic energy, becomes

$$E^{\ell}_{J} = \frac{1}{2}\left[U_J - \frac{\bar{\psi}^{\ell}_{J+1} - \bar{\psi}^{\ell}_{J-1}}{2\delta y}\right]^2 \tag{4.21}$$

The normalized sum is, by the trapezoidal rule

$$E' = \frac{1}{2}\left\{E'_1 \frac{\delta y}{2} + \sum_{J=2}^{N-1} E'_J \, \delta y + E'_N \frac{\delta y}{2}\right\}$$
$$= \frac{\delta y}{2} \sum_{H=2}^{N-1} E'_J = \frac{1}{N-1} \sum_{J=2}^{N-1} E'_J \tag{4.22}$$

Similarly, equation (2.31) for the kinetic energy in the turbulent motion becomes,

$$\hat{E}'_{K,J} = \frac{1}{2}\left\{\left[\frac{\hat{\Psi}'_{K,J+1} - \hat{\Psi}'_{K,J-1}}{2\delta y}\right]^2 + \left[\frac{\hat{\Psi}'_{K+1,J} - \hat{\Psi}'_{K-1,J}}{2\delta x}\right]^2\right\} \tag{4.23}$$

The associated mean value is

$$\hat{E}'_J = \frac{1}{M} \sum_{K=1}^{M} \hat{E}'_{K,J} \tag{4.24}$$

And the normalized sum over the domain is

$$\hat{E}' = \frac{1}{N-2} \sum_{J=2}^{N-1} \hat{E}'_J \tag{4.25}$$

Equation (2.34) for the Reynolds stress becomes

$$S'_J = \frac{1}{M} \sum_{K=1}^{M} \left\{\frac{\hat{\Psi}'_{K+1,J} - \hat{\Psi}'_{K-1,J}}{2\delta x} \frac{\hat{\Psi}'_{K,J+1} - \hat{\Psi}'_{K,J-1}}{2\delta y}\right\} \tag{4.26}$$

C. CONVERGENCE OF THE NUMERICAL SOLUTION

A rigorous proof of the convergence of the numerical solution to the solution of the non-linear partial differential equation is not possible. Even if the non-linear terms are ignored in the analysis, certain difficulties remain. However, reasonable assurance of convergence can be obtained by analysis of the linear equations, and by numerical experiments. Finally, the convergence and stability of the

scheme, in the linear range, can be demonstrated by calculations which employ as initial conditions the eigenfunctions of the Orr-Sommerfeld equation.

Richtmyer (1967) has shown that, for certain properly posed, linear initial value problems, a finite difference approximation of the partial differential equation which is both consistent and stable, is necessarily convergent. A finite difference approximation is consistent with the differential equation if the associated truncation error tends to zero, as the magnitudes of the time and space increments tend to zero. It is stable if the growth of errors in the numerical solution is bounded.

The consistency of the finite-difference equations is considered in Section C.1. An analysis of the stability of an analogous linear difference equation is given in Section C.2. Finally, in Section C.3., the stability of the complete non-linear system is considered.

1. Consistency

The truncation error in a finite difference approximation is obtained by means of Taylor Series expansions. The results for typical terms in equations (4.3) and (4.4) are

$$\frac{\delta\psi}{\delta x} = \frac{\partial\psi}{\partial x} + \frac{\delta x^2}{6}\left(\frac{\partial^3\psi}{\partial x^3} + \frac{\delta x^2}{20}\frac{\partial^5\psi}{\partial x^5} + \cdots\right.$$

$$\frac{\delta^2\psi}{\delta x^2} = \frac{\partial^2\psi}{\partial x^2} + \frac{\delta x^2}{12}\left(\frac{\partial^4\psi}{\partial x^4} + \frac{\delta x^2}{30}\frac{\partial^6\psi}{\partial x^6} + \cdots\right.$$

$$J_{K,J} = J(\psi,\Gamma) + \frac{\delta x^2}{6}\left[\frac{\partial\psi}{\partial y}\frac{\partial^3\Gamma}{\partial x^3} + 3\frac{\partial}{\partial x}\left(\frac{\partial\Gamma}{\partial x}\frac{\partial^2\psi}{\partial x\partial y} - \frac{\partial\psi}{\partial x}\frac{\partial^2\Gamma}{\partial x\partial y}\right) - \frac{\partial\Gamma}{\partial y}\frac{\partial^3\psi}{\partial x^3}\right]$$

$$+ \frac{\delta y^2}{6}\left[\frac{\partial\Gamma}{\partial x}\frac{\partial^3\psi}{\partial y^3} + 3\frac{\partial}{\partial y}\left(\frac{\partial\psi}{\partial y}\frac{\partial^2\Gamma}{\partial x\partial y} - \frac{\partial\Gamma}{\partial y}\frac{\partial^2\psi}{\partial x\partial y}\right) - \frac{\partial\psi}{\partial x}\frac{\partial^3\Gamma}{\partial y^3}\right]$$

$$+ \cdots$$

$$D^{t}_{K,J} = \frac{\partial^2\Gamma}{\partial x^2} + \frac{\partial^2\Gamma}{\partial y^2} - \delta t^2 \left(\frac{\delta x^2 + \delta y^2}{\delta x^2\, \delta t^2}\right)\left[\frac{\partial^2\Gamma}{\partial t^2} + \frac{\delta t^2}{12}\left(\frac{\partial^4\Gamma}{\partial t^4} + \frac{\delta t^2}{30}\frac{\partial^6\Gamma}{\partial t^6} + \cdots\right.\right.$$

$$+ \frac{\delta x^2}{12}\left(\frac{\partial^4\Gamma}{\partial x^4} + \frac{\delta x^2}{30}\frac{\partial^6\Gamma}{\partial x^6} + \cdots\right.$$

$$+ \frac{\delta y^2}{12}\left(\frac{\partial^4\Gamma}{\partial y^4} + \frac{\delta x^2}{30}\frac{\partial^6\Gamma}{\partial y^6} + \cdots\right.$$

Gathering terms we find that, to the second order, the total truncation error in equation (4.3) is

$$- \frac{\delta t^2}{R}\left(\frac{\delta x^2 + \delta y^2}{\delta x^2\, \delta y^2}\right)\frac{\partial^2\Gamma}{\partial t^2} - \frac{\delta t^2}{6}\left[\frac{\partial^3\Gamma}{\partial t^3} + \frac{\delta t^2}{2R}\left(\frac{\delta x^2 + \delta y^2}{\delta x^2\, \delta y^2}\right)\frac{\partial^4\Gamma}{\partial t^4}\right]$$

$$+ \frac{\delta x^2}{6}\left[- U_J \frac{\partial^3\Gamma}{\partial x^3} + U'' \frac{\partial^3\psi}{\partial x^3} + \frac{\partial\psi}{\partial y}\frac{\partial^3\Gamma}{\partial x^3} + 3\frac{\partial}{\partial x}\left(\frac{\partial\Gamma}{\partial x}\frac{\partial^2\psi}{\partial x\partial y} - \frac{\partial\psi}{\partial x}\frac{\partial^2\Gamma}{\partial x\partial y}\right)\right.$$

$$\left. - \frac{\partial\Gamma}{\partial y}\frac{\partial^3\psi}{\partial x^3} + \frac{1}{2R}\frac{\partial^4\Gamma}{\partial x^4}\right] \qquad (4.27)$$

$$+ \frac{\delta y^2}{6}\left[\frac{\partial\Gamma}{\partial x}\frac{\partial^3\psi}{\partial y^3} + 3\frac{\partial}{\partial y}\left(\frac{\partial\psi}{\partial y}\frac{\partial^2\Gamma}{\partial x\partial y} - \frac{\partial\Gamma}{\partial y}\frac{\partial^2\psi}{\partial x\partial y}\right) - \frac{\partial\psi}{\partial x}\frac{\partial^3\Gamma}{\partial y^3} + \frac{1}{2R}\frac{\partial^4\Gamma}{\partial x^4}\right]$$

Thus the truncation error contains terms proportional to

$$\frac{\delta t^2}{R}\left(\frac{\delta x^2 + \delta y^2}{\delta x^2\, \delta y^2}\right),\ \delta t^2,\ \delta x^2 \text{ and } \delta y^2$$

The first terms arises from the Du Fort-Frankel approximation. Compatibility requires that $\delta t \to 0$ faster than δx or δy. The truncation error in equation (4.4) is

$$\frac{\delta x^2}{12}\frac{\partial^4\psi}{\partial x^4} + \frac{\delta y^2}{12}\frac{\partial^4\psi}{\partial y^4} \qquad (4.28)$$

Similarly the truncation error associated with the second-order formulation of the vorticity boundary condition is

$$\frac{\delta y^2}{6} \frac{\partial^4 \psi}{\partial y^4} \tag{4.29}$$

It should be noted that the approximation for the boundary values of the vorticity involves values of the stream function at time level 1. This approximation is correctly centered with respect to the time differencing which involves values at 1+1 and 1-1.

The over-all order of the approximation is

$$0\left[\frac{\delta t^2}{R}\left(\frac{\delta x^2 + \delta y^2}{\delta x^2\ \delta y^2}\right) + \delta t^2 + \delta x^2 + \delta y^2\right]$$

and the scheme is consistent if

$$\left.\begin{matrix}\delta x \\ \delta y \\ \delta t\end{matrix}\right\} \rightarrow 0 \qquad \text{LIM} \quad \frac{\delta t^2}{R}\left(\frac{\delta x^2 + \delta y^2}{\delta x^2\ \delta y^2}\right) = 0 \tag{4.30}$$

2. Stability Analysis of an Analogous Linear Equation

The numerical solution of the finite difference equations is stable if the round-off errors due to the finite decimal representation of real numbers are bounded in their growth. In the case of linear equations, the errors introduced at a given step are propagated by the same system of finite difference equations. In addition, the linearity permits, in the analysis, the superposition of random errors introduced at each step. For the purpose of determining what constraints, if any, must obtain for stability, it is only necessary to consider the amplification, by the equations, of errors introduced at the initial or an arbitrary step.

Two approaches are in wide use. The first involves the determination of the spectral radius of the matrix which defines the time step (Smith 1965). The finite-difference equations are expressed in the form

$$\bar{E}_t = A\bar{E}_{t-1} \tag{4.31}$$

Where $\bar{E}_t$ is a vector whose components are spatial values of the dependent variable, (or in this analysis, the errors), at time step t. If the matrix A, of dimension σ, has σ linearly independent eigenvectors, $\bar{\eta}_s$, then an arbitrary initial error can be expressed in terms of them.

$$\bar{E}_0 = \sum_{s=1}^{\sigma} c_s \bar{\eta}_s$$

It follows from equation (3.39) that

$$\bar{E}_t = A^t \bar{E}_0$$

Thus if ν_s are the eigenvalues of A then

$$\bar{E}_t = \sum_{s=1}^{\sigma} c_s \nu_s^t \bar{\eta}_s \tag{4.32}$$

Thus $\bar{E}_t$ is bounded if the largest of the modulii of ν_s is less than unity. In some fairly simple cases, which generally involve two-level finite-difference formulas such as Forward or Modified Euler differencing, the bounds can be determined in a straightforward way by application of theorems due to Gerschgorin and Brauer (Smith 1965).

If multi-level formulas, such as central time differencing are employed then the expression of the finite difference equations in the form of equation (4.31) is not so easy. In addition the determination of the eigenvalues or the spectral radius of the resulting matrix can be a formidable task.

A second approach which generally offers fewer difficulties, but which, unlike the matrix approach, does not take into account the effect of the boundary conditions is due to von Neumann. It was introduced in a paper by O'Brien, Hyman and Kaplan (1951), and is employed here. The initial errors are expressed in terms of their Fourier components. If none of the modes grow with increasing time the scheme is stable. Because the equations considered in the analysis are linear the various modes do not interact, and we need only to consider the behavior of an arbitrary mode.

We consider a finite difference equation for the initial errors which is linear, but analogous to equation (4.7).

$$\frac{E^{t+1}_{K,J} - E^{t-1}_{K,J}}{2\delta t} = u_0 \frac{E^{t}_{K+1,J} - E^{t}_{K-1,J}}{2\delta x} + v_0 \frac{E^{t}_{K,J+1} - E^{t}_{K,J-1}}{2\delta y}$$

$$+ \frac{1}{R}\left[\frac{E^{t}_{K+1,J} - \left(E^{t+1}_{K,J} + E^{t-1}_{K,J}\right) + E^{t}_{K-1,J}}{\delta x^2}\right. \tag{4.33}$$

$$\left. + \frac{E^{t}_{K,J+1} - \left(E^{t+1}_{K,J} + E^{t-1}_{K,J}\right) + E^{t}_{K-1,J}}{\delta y^2}\right]$$

Let

$$r_x = u_0 \frac{\delta t}{\delta x} \qquad r_y = v_0 \frac{\delta t}{\delta y}$$

$$\lambda_x = \frac{2\delta t}{R\delta x^2} \qquad \lambda_y = \frac{2\delta t}{R\delta y^2} \qquad \lambda = \lambda_x + \lambda_y \tag{4.34}$$

Then equation (4.33) becomes (4.35)

$$(1+\lambda)\, E^{t+1}_{K,J} = (1-\lambda)\, E^{t-1}_{K,J} + r_x\left(E^{t}_{K+1,J} - E^{t}_{K-1,J}\right) + r_y\left(E^{t}_{K,J+1} - E^{t}_{K,J-1}\right)$$

$$+ \lambda_x\left(E^{t}_{K+1,J} + E^{t}_{K-1,J}\right) + \lambda_y\left(E^{t}_{K,J+1} + E^{t}_{K,J-1}\right)$$

Let

$$E^{t}_{K,J} = \sum_{n=0}^{N} \sum_{m=0}^{M} A_{nm}\, e^{\alpha nm t}\, e^{i \frac{n\pi x}{h} K}\, e^{i\, m\pi y J}$$

$$= \sum_{n=0}^{N} \sum_{m=0}^{M} A_{nm}\, \xi^{t}_{nm}\, e^{i\, \gamma_x nK \;\; i\, \gamma_y mJ} \tag{4.36}$$

Where A_{nm} are determined from the specified initial errors and $\xi_{nm} = e^{\alpha nm \delta t}$.

For stability it is required that

$$|\xi_{nm}| \leq 1$$

Since for the linear equation we need only to consider an arbitrary combination of spatial modes, the substitution

$$E^{t}_{K,J} = \xi^{t}\, e^{i\, \gamma_x K}\, e^{i\, \gamma_y J}$$

in equation (4.35) will reveal the necessary constraints. After cancelling the common factor $\xi^{t-1}\, e^{i\, \gamma_x K}\, e^{i\, \gamma_y J}$ and rearranging, equation (4.35) becomes

$$\xi^2 - \frac{2}{1+\lambda}\left[(\lambda_x \cos \gamma_x + \lambda_y \cos \gamma_y) + i\,(r_x \sin \gamma_x + r_y \sin \gamma_y)\right]\xi$$

$$- \frac{1-\lambda}{1+\lambda} = 0 \tag{4.37}$$

Which has roots

$$\xi = \frac{1}{1+\lambda}\left\{(\lambda_x \cos \gamma_x + \lambda_y \cos \gamma_y) + i\,(r_x \sin \gamma_x + r_y \sin \gamma_y)\right.$$

$$\left. \pm \sqrt{\left[(\lambda_x \cos \gamma_x + \lambda_y \cos \gamma_y) + i(r_x \sin \gamma_x + r_y \sin \gamma_y)\right]^2 + (1-\lambda^2)} \right\} \tag{4.38}$$

Fromm (1963) has performed the analysis for the case $\delta x = \delta y$, which in the present notation means

$$r_x = r_y$$

$$\lambda_x = \lambda_y \qquad \lambda = 2\lambda_x$$

Because of the obvious difficulty in determining bounds on the modulus of ξ in equation (4.38), Fromm performed the analysis in two steps. In the first step only the diffusion terms are treated: the transport terms are assumed to be zero. In the second step the procedure is reversed. The following is an extention of Fromm's analysis for the case $\delta x \neq \delta y$.

Let

$$\omega = \lambda_x \cos \gamma_x + \lambda_y \cos \gamma_y$$

$$\Omega = r_x \sin \gamma_x + r_y \sin \gamma_y$$

Equation (4.38) becomes

$$\xi = \frac{1}{1+\lambda}\left\{\omega + i\Omega \pm \sqrt{(\omega+i\Omega)^2 + 1-\lambda^2}\right\} \tag{4.39}$$

For diffusion with no transport $\Omega = 0$.

Equation (4.39) becomes

$$\xi = \frac{1}{1+\lambda}\left\{\omega \pm \sqrt{\omega^2 + 1-\lambda^2}\right\} \tag{4.40}$$

Note that

$$-\lambda \leq \omega \leq \lambda$$

If the radical in (4.40) is real then $|\xi|$ reaches a maximum for $\omega^2 = \lambda^2$ where

$$\xi = \pm \frac{(\lambda+1)}{(\lambda+1)} = \pm 1$$

If the radical is imaginary then

$$\xi = \frac{1}{1+\lambda}\left\{ \omega \pm i \sqrt{\lambda^2-\omega^2-1} \right\}$$

$$|\xi| = \frac{\sqrt{\lambda^2-1}}{1+\lambda} = \sqrt{\frac{\lambda-1}{\lambda+1}}$$

In each case $|\xi| \leq 1$ for all values of λ_x and λ_y. Thus, in relation to the diffusion terms, the Du Fort-Frankel Central differencing is stable at all values of δt. This result is given by Richtmyer (1967) for the one-dimensional case.

For transport with no diffusion, $\omega = 0$, and $\lambda = 0$. Equation (4.39) becomes

$$\xi = i\Omega \pm \sqrt{-\Omega^2 + 1} \qquad (4.41)$$

If $\Omega^2 \leq 1$ the radical is real and

$$|\xi| = 1$$

If, however, $\Omega^2 > 1$, the radical is imaginary and

$$\xi = i\left(-\Omega \pm \sqrt{\Omega^2 - 1}\right)$$

$$|\xi| = 2\Omega^2 - 1 \mp 2\Omega\sqrt{\Omega^2-1}$$

For the second root

$$|\xi| > 1$$

Thus, in relation to the transport terms the Central differencing is stable if

$$\Omega^2 = (r_x \sin \gamma_x + r_y \sin \gamma_y)^2 \leq 1$$

Then from equations (4.34)

$$\delta t \left| \frac{u_0}{\delta x} + \frac{v_0}{\delta y} \right| \leq 1$$

A more conservative form is

$$\delta t \left(\frac{|u_0|}{\delta x} + \frac{|v_0|}{\delta y} \right) \leq 1$$

which is equivalent to

$$\delta t \leq \frac{\delta x \delta y}{|u_0| \delta y + |v_0| \delta x} \tag{4.42}$$

Although a complete analysis of even the linear equations is not carried out it is reasonable to expect that a constraint of the form of equation (4.42) will be required.

3. Stability and Accuracy of the Complete Non-linear Equations

From the stability analysis of the linear equations it was expected that a constraint of the form of equation (4.42) would be required for the general case where both diffusion and advection terms are present, the latter including the non-linear terms. Accordingly, the value of the time increment was determined by

$$\delta t = \frac{f \, \delta x \, \delta y}{u_0 \, \delta y + v_0 \, \delta x} \tag{4.43}$$

where u_0 and v_0 are the maximum absolute values of the x and y components of total velocity over a large sample of the mesh points, and f is a parameter which was determined by numerical experiment. The values of u_0 and v_0 were recomputed at regular intervals, and δt was adjusted accordingly. Experiments with a variety of mesh sizes and disturbance amplitudes indicated critical values of the parameter f in the range $.95 < f_c < 1.02$. When the critical value of f was exceeded,

the instabilities developed quite rapidly, always within 100 time steps and in most cases within 50 steps. At low disturbance amplitudes the instability developed first in the midstream region, where the laminar transport velocity is the greatest. At higher amplitudes of the disturbance, the instability occurred first in the wall region where, owing to the character of the disturbance, the advection was the greatest. The instabilities were, in most cases, unmistakable: an explosive growth in the numerical values of the disturbance quantities immediately preceded an abnormal termination of the calculation due to numerical overflow. At values of f very close to the critical, a more insidious situation occurred in which the initial amplification of errors in the disturbance stream function and vorticity, and hence in the computed quantities u_0 and v_0, was inhibited by the ensuing automatic reduction of δt. The calculations were characterized by periodic amplification and decay of large errors. The results were in poor agreement with recalculations with a reduced value of δt. If the automatic recalculation of the time step was bypassed then the explosive instability described above would occur. Accordingly f was limited to values less than .6, where no such difficulties were encountered.

In similar numerical experiments with the Du Fort-Frankel Central differencing applied to the non-linear vorticity transport equations, Fromm determined a critical value of f equal to .8. An important difference between the finite-difference equations employed by Fromm and those in the present work is associated with the treatment of the non-linear terms. Fromm employed a representation of those terms equivalent to $JB_{K,J}$ of equations (3.6). As has been

noted, in the present case the mean of $JA_{K,J}$, $JB_{K,J}$, and $JC_{K,J}$, is employed. The significance of the various finite difference representations of the non-linear terms is considered below.

Phillips (1959) has shown the existence of numerical instabilities, associated with the non-linear terms, which result from the discrete representation of continuous functions, but which, in principle, cannot always be eliminated by reducing the time step or the mesh spacing. This kind of instability is the result of uncontrolled "aliasing" or confusion of frequencies (Hamming 1962). To illustrate, note that at a given value of the index J the M values

$$\psi_{K,J} \qquad K = 1,2,\cdots M$$

can be expressed in terms of $\frac{M}{2} + 1$ distinct Fourier modes.

$$\psi_{K,J} = \frac{1}{M}\left\{a_{0,J} + 2\sum_{n=1}^{M/2-1}\left(a_{n,J}\cos\frac{2n\pi K}{M} + b_{n,J}\sin\frac{2n\pi K}{M}\right) + a_{M/2,J}\right\}$$

where the period associated with the highest mode number is $2\delta x$. Since,

$$\cos\frac{2n\pi K}{M}\cos\frac{2n\pi K}{M} = \frac{1}{2}\cos\frac{2(n+m)\pi K}{M} + \cos\frac{2(n-m)\pi K}{M} \qquad \text{etc.,}$$

the non-linear interaction between modes n and m will produce excitation in modes n+m and n-m. If n+m is greater than M/2 that interaction will be misrepresented by the discrete system.

Let $m + n = \frac{M}{2} + r$, $r < \frac{M}{2}$

$$\cos\frac{2(M/2 + r)\,\pi K}{M} = \cos\frac{2r\pi K}{M}$$

$$\sin\frac{2(M/2 + r)\,\pi K}{M} = -\sin\frac{2r\pi K}{M}$$

Thus the aliasing involves a spurious interaction with mode $r = n+m - M/2$. If both m and n are less than M/4, that is if each wave has a period greater than $4\delta x$, no misrepresentation will occur. The instability that results from aliasing occurs after a large number of time steps and is characterized by explosive growth of the energy in the system. Phillips was able to suppress this instability by, "periodically eliminating all Fourier components with wave lengths smaller than $4\delta x$", that is, by zeroing the upper half of the representable spectrum. In effect, in each spatial or spectral coordinate, twice as many degrees of freedom are carried in the calculations as are retained in the solution. For calculations explicitly in terms of the spectral form of the equations this seems to be the only recourse.

Arakawa (1966) has devised a number of finite difference forms for the non-linear advection terms which eliminate this instability by controlling or limiting the aliasing. They are various linear combinations of the forms JA, JB and JC as given, in terms of the central difference operators, by equations (4.6). For a confined flow or one with identical conditions at inflow and outflow boundaries it is easy to show that, in the absence of viscosity, the spatial integrals of kinetic energy and any function of vorticity are fixed quantities. In a viscous flow the advection terms make no contribution to the rates of change of these quantities. Lilly (1965) has shown that the feature essential to the control of the aliasing is the conservation of mean kinetic energy or mean squared vorticity implicit in the finite-difference representation of the advection terms. A number of stable conservative schemes are given by both Arakawa and Lilly. The only scheme which conserves both mean kinetic energy and squared vorticity is

$$J(\psi,\Gamma) = \frac{1}{3}\ (JA + JB + JC)$$

which is the form used in the present undertaking.

There is, in addition, a third type of instability which is associated with the central time differencing. This weak instability is a computational mode associated with the three-time level central difference representation of $\frac{\partial \Gamma}{\partial t} = f(\Gamma)$. In effect, the central difference representation replaces a first order (in time) continuous equation with a second order discrete one, which has an additional, non-physical, solution (Lilly 1965). It takes the form of a slowly growing oscillation, of period $2\delta t$, about the true solution. If it remains unchecked then eventually the solutions at even and odd time steps diverge. This computational mode can be suppressed quite simply by periodically averaging the solutions over successive time steps and restarting the calculations with the averaged values (Williams 1969).

This periodic interruption of the sequence of the calculation provides a convenient opportunity for the readjustment of the value of δt. In the midst of the continued central time differencing δt must remain fixed. However when the averaging takes place it can be adjusted in accordance with equation (4.43), a single time step performed by the Modified Euler method (which does not require values at the previous time level), and the normal sequence can be continued. This procedure proved to be quite effective for both purposes. No evidence of the growth of the computational mode was observed.

Fischer (1965) tested the accuracy of four methods of time differencing on a one-dimensional linearized version of the equations of motion, and found that the central differencing was decidedly

superior. In similar tests on the wave equation Kurihara (1965) found that, provided the computational mode is suppressed, it was superior in terms of measured errors in amplitude and phase velocity to the other methods tested.

D. SOLUTION OF POISSON'S EQUATION FOR THE STREAM FUNCTION

The finite-difference relation between the stream function and the vorticity is given by equation (4.8). In the course of each time step new values of the vorticity are obtained from equation (4.7). Equation (4.8) must then be solved for the matching values of the stream function.

$$\Gamma_{K,J} = \frac{\psi_{K+1,J} - 2\psi_{K,J} + \psi_{K,J-1}}{\delta x^2} + \frac{\psi_{K,J+1} - 2\psi_{K,J} + \psi_{K,J-1}}{\delta y^2} \qquad (4.8)$$

$$K = 1,2,\cdots M$$

$$J = 2,3,\cdots N-1$$

In matrix form equations (4.8) are

$$\bar{\Gamma} = B\bar{\psi} \qquad (4.44)$$

where $\bar{\Gamma}$ and $\bar{\psi}$ are vectors of dimension $d = (N-2)\,M$ and B is a banded symmetric matrix of dimension d x d. The solution to equation (4.44),

$$\bar{\psi} = B^{-1}\bar{\Gamma}$$

is obtained in three steps by means of the discrete Fourier transform. In the transformed domain the finite-difference relation between the stream function and the vorticity has the form

$$\bar{\Gamma}_n = A_n \bar{\psi}_n \qquad n = 1,2,\cdots\frac{M}{2} + 1 \qquad (4.45)$$

where $\bar{\Gamma}_n$ and $\bar{\psi}_n$ are vectors of dimension N-2 and the A_n are symmetric tridiagonal matrices of dimension (N-2) x (N-2). Because the matrices

A_n are tridiagonal, the solutions to equations (4.45) can be obtained by a particularly simple case of Gauss elimination (Richtmyer 1967).

The discrete Fourier transform pair is given by

$$T_n = \sum_{K=1}^{M} F_K e^{i\theta_{n,K}} \qquad n = 1,2,\cdots M$$
$$F_K = \frac{1}{M} \sum_{n=1}^{M} T_n e^{-i\theta_{n,K}} \qquad K = 1,2,\cdots M \tag{4.46}$$

Where T_n and F_K are, in general, complex and,

$$\theta_{n,K} = \frac{2\pi(n-1)\ (K-1)}{M} \tag{4.47}$$

The associated orthogonality relation is

$$\frac{1}{M} \sum_{n=1}^{M} e^{i\theta_{n,s-K+1}} = 1 \qquad \text{if } s=k$$
$$= 0 \qquad \text{otherwise} \tag{4.48}$$

For the solution of equation (4.8) let

$$G_{n,J} = \sum_{K=1}^{M} \Gamma_{K,J} e^{i\theta_{n,K}} \qquad K = 1,2,\cdots M$$
$$F_{n,J} = \sum_{K=1}^{M} \psi_{K,J} e^{i\theta_{n,K}} \qquad J = 1,2,\cdots N \tag{4.49a}$$

Then from equations (3.65) $\Gamma_{K,J}$ and $\psi_{K,J}$, are given by

$$\Gamma_{K,J} = \frac{1}{M} \sum_{n=1}^{M} G_{n,J} e^{-i\theta_{n,K}} \qquad K = 1,2,\cdots M$$
$$\psi_{K,J} = \frac{1}{M} \sum_{n=1}^{M} F_{n,J} e^{-i\theta_{n,K}} \qquad J = 1,2,\cdots N \tag{4.49b}$$

From equations (3.1) and (3.2) it is evident that

$$\theta_{n,K} = \frac{(x_K - c)\ 2\pi\ (n-1)}{2h}$$

where

$$c = \frac{\delta x}{2} - h$$

Therefore equations (4.48) and (4.49) implicitly satisfy the periodic boundary conditions on Γ and ψ. The no-slip boundary conditions require that

$$F_{n,1} = F_{n,N} = 0 \qquad n = 1,2,\cdots M$$

Substitution of equations (4.49) in equation (4.8) yields,

$$\sum_{n=1}^{M} \left\{ \delta y^2 \, G_{n,J} - \left[F_{n,J+1} - \alpha_n F_{n,J} + F_{n,J+1} \right] \right\} e^{-i \theta_{n,K}} = 0 \qquad (4.50)$$

$$K = 1,2,\cdots M$$

$$J = 2,3,\cdots N-1$$

where

$$\alpha_n = 2 \left[1 - \frac{\delta y^2}{\delta x^2} (\cos \theta_{n,2} - 1 \right] \qquad (4.51)$$

Equation (4.50) can be satisfied at all values of K only if

$$\delta y^2 \, G_{n,J} = F_{n,J+1} - \alpha_n F_{n,J} + F_{n,J+1} \qquad (4.52)$$

$$n = 1,2,\cdots M$$

$$J = 2,3,\cdots N-1$$

Although $F_{n,J}$ and $G_{n,j}$ are complex, the coefficients in equation (4.52) are all real. The solutions for the real and imaginary parts may be obtained separately. Because $\Gamma_{K,J}$, and $\psi_{K,J}$, are real, the real and imaginary parts of $G_{n,J}$, and $F_{n,J}$, are symmetric and anti-symmetric respectively about $n = \frac{M}{2} + 1$. (Recall that M, as defined, is always even.) Thus M x (N-2) quantities in the x domain beget exactly M x (N-2) distinct quantities in the wave number domain. There are

exactly M distinct sets of equations of the form equation (4.52) in terms of real variables.

Equations (4.8) can be solved in this manner without the heavy storage penalties that are usually associated with direct methods. In this case the additional storage requirements are trivial: an M dimensional complex vector for the row by row transformation of $\Gamma_{K,J}$ and the inverse transformation of $\psi_{K,J}$, and three N-2 dimensional vectors to accommodate the solution of the tridiagonal systems. The vorticity array is transformed row by row, and the transformed quantities are stored in the array reserved for the stream function. The tridiagonal equations (4.52) are solved column by column. In each step, the solution for the transform of the stream function replaces the stored values of the transform of the vorticity which are no longer required. Finally the transform of the stream function is inverted row by row. In order to minimize the accumulation of roundoff errors in the direct solution, the calculations associated with both the discrete transform and the Gauss elimination are performed in double precision. The equivalent of sixteen decimal digits are carried in the calculations.

The transforms and inverse transforms are obtained by means of the remarkably efficient Fast Fourier Transform algorithm of Cooley and Tukey (1965). The computational effort required to perform the transform on an M dimensional vector is proportional to M log (M). There are roughly 2 N such transforms required in the solution of equation (4.8). The effort required to solve each of the M tridiagonal systems is proportional to 3 N. Hence the total computational effort is proportional to NM (2 log (M) + 3).

In an early version of the program equation (4.8) was solved by Successive Over-relaxation. It is an iterative method which is easy to program, involves no storage penalties and which, at least for small arrays, is rapidly convergent.

Equation (4.8) can be written

$$\psi_{K,J} = \frac{\delta y^2}{2(\delta x^2+\delta y^2)}\left[\psi_{K+1,J}+\psi_{K-1,J}+\frac{\delta x^2}{\delta y^2}\left(\psi_{K,J+1}+\psi_{K,J-1}\right)-\delta x^2\,\Gamma_{K,J}\right] \tag{4.53}$$

Equation (4.53) can be used as the basis for various iterative formulations. If the superscript denotes the iteration number then

$$\psi^{t+1}_{K,J} = \frac{\delta y^2}{2(\delta x^2+\delta y^2)}\left[\psi^{t}_{K+1,J}+\psi^{t}_{K-1,J}+\frac{\delta x^2}{\delta y^2}\left(\psi^{t}_{K,J+1}+\psi^{t}_{K,J-1}\right)-\delta x^2\,\Gamma_{K,J}\right] \tag{4.54}$$

defines a Jacobi iteration. With this method separate computer storage must be reserved for the t and $t+1$ iterates. If the current values are employed in the formula as soon as they are available then the extra storage requirement is eliminated. In addition the convergence rate is, for the Poisson equation, approximately doubled (Varga 1962). Equation (4.54) becomes

$$\psi^{t+1}_{K,J} = \frac{\delta y^2}{2(\delta x^2+\delta y^2)}\left[\psi^{t}_{K+1,J}+\psi^{t+1}_{K-1,J}+\frac{\delta x^2}{\delta y^2}\left(\psi^{t}_{K,J+1}+\psi^{t+1}_{K,J-1}\right)-\delta x^2\,\Gamma_{K,J}\right] \tag{4.55}$$

Equation (4.55) defines a Gauss-Seidel iteration.
By the addition of an acceleration or over-relaxation parameter to equation (4.55) the convergence rate can be considerably increased

$$\psi^{t+1}_{K,J} = (1-\omega)\,\psi^{t}_{K,J} + \frac{\omega\delta y^2}{2(\delta x^2+\delta y^2)}\left[\psi^{t}_{K+1,J}+\psi^{t+1}_{K-1,J}+\frac{\delta x^2}{\delta y^2}\left(\psi^{t}_{K,J+1}+\psi^{t+1}_{K,J-1}\right)-\delta x^2\Gamma_{K,J}\right] \tag{4.56}$$

Equation (4.56) defines the Successive Over-relaxation. In this case the convergence rate is strongly dependent on the choice of ω. The optimum value of ω, which for the Poisson equation falls between one and two, is determined by the modulus of the largest eigenvalue of the matrix which defines the system of equations (Varga 1962), however it is generally easier to determine it by experiment.

With each of the various iterative methods an approximate solution to the system of equations is obtained by systematically applying the iteration formula at each point in the field until some measure of the changes in the various values falls below a specified tolerance. In order to equalize the influence of the boundary conditions it is necessary to start successive sweeps at opposite boundaries.

Early experiments with certain contrived test problems indicated that with the optimum value of ω, convergence was generally obtained with one fifth the number of iterations required for the Gauss-Seidel Iteration, and that the errors in the solution were reduced by as much as a factor of ten. Unfortunately in application to the present problem the performance of even the Over-relaxation is inadequate. The computational effort required to perform one iteration in an M by N array is proportional to M N. The number of iterations required for convergence to a fixed tolerance was roughly proportional to $(M\ N)^{1/4}$. Finally the tolerance required for a fixed error in the solution varied nearly inversely as $(M\ N)^{1/2}$. The over-all computational effort was roughly proportional to $(M\ N)^{7/4}$. In the integration of the vorticity transport equations the relaxation solution to the Poisson equation required more than 80 per cent of the total computing time. More sophisticated relaxation techniques, such as line or block over-relaxation,

might double the convergence rate (Varga 1962), but the essential difficulties remain for sufficiently large dimensions.

A number of numerical tests were conducted in order to compare the efficiency and accuracy of the various schemes. In these tests values of $\psi_{K,J}$ were established from

$$\psi_{K,J} = \left(1 - Y_J^2\right)^2 \left\{ Y_J^2 + \sum_{m=1}^{6} 1/m^2 \left[\sin\,(mx_K) + Y_J^2 \cos\,(mx_K)\right]\right\}$$

Values of $\Gamma_{K,J}$, were obtained from the prescribed values of $\psi_{K,J}$ by application of equation (4.8). Thus the stored values of $\psi_{K,J}$, are an exact solution of the finite-difference equations (4.8) corresponding to the values of $\Gamma_{K,J}$. Solutions of equations (4.8) denoted by $s_{K,J}$, were obtained by Gauss-Seidel iteration, Successive Over-relaxation and by the direct method.

The mean and maximum relative errors defined by,

$$E_{mean} = \frac{1}{M(N-2)} \sum_{J=2}^{N-1} \sum_{K=1}^{M} \left| \frac{s_{K,J} - \psi_{K,J}}{\psi_{K,J}} \right|$$

$$E_{max} = \max \left\{ \left| \frac{s_{K,J} - \psi_{K,J}}{\psi_{K,J}} \right| , \quad \begin{matrix} K = 1,2,\cdots M \\ J = 2,3,\cdots N-1 \end{matrix} \right\}$$

were calculated for each test.

The iterative processes were assumed to have converged when at every point in the field

$$\frac{\left| s_{K,J}^{t+1} - s_{K,J}^{t} \right|}{\bar{s}} < \epsilon$$

where $\bar{s}$ is the mean absolute value of $s_{K,J}$, at the start of the iteration procedure and ϵ is an assigned tolerance. In order to minimize the computer time expended in testing for convergence, the

test was applied only at even-numbered sweeps. In addition the test was bypassed at all points following the first point at which it was not met. Initial values of $s_{K,J}$, were established reasonably close to the exact solution in order to approximate the situation in the numerical integration, where values of $\psi_{K,J}$ at time t are used as a first guess for values at $t + \delta t$ in the iterative procedure. These initial values were

$$s^{0}_{K,J} = \psi_{K,J} + \frac{1}{10}\left[1 - |Y_J|\right]\left[\sin(2x_K) + \cos(5x_K)\right]$$

In every case the experimentally determined optimum value of the relaxation parameter, ω, was employed for the Successive Over-relaxation.

Internal timers were started and stopped at the beginning and at the end of each solution and the elapsed times, in seconds, were printed along with errors. The results are given in Table II. They indicate rather dramatically the superiority of the direct solution in terms of both accuracy and speed.

Tests with the iterative methods were performed at two and three values of the convergence tolerance in order to verify that the procedures were still converging systematically at the specified tolerances. Generally a reduction of the convergence tolerance by a factor of two produced a solution with the errors reduced by about the same factor. The effects of the rapid reduction of the convergence rates of the iterative methods due to increased array dimensions are quite evident.

The Gauss-Seidel iteration is, in particular, quite useless in application to large arrays. For the 64 by 201 point mesh, a comparison of the tests at $\varepsilon = 10^{-4}$ and 5×10^{-5} shows that the expenditure of

	Gauss Seidel Iteration			Successive Over-Relaxation			Fourier Trans-form Solution	M N ω
ε	.0002	.0001	.00005	.0002	.0001	.00005		
Iterations		214	260		58	66		
Time		20.4	24.05		5.8	6.6	.9	32x41
Mean Error		.017	.0086		.0025	.0010	.00006	
Maximum Error		2.262	1.159		.4747	.1980	.00094	ω=1.785
Iterations		484	666		104	120		
Time		179.1	245.8		41.0	48.0	3.8	64x81
Mean Error		.1646	.0820		.0065	.0040	.0004	
Maximum Error		238.4	117.3		3.393	1.772	.2104	ω=1.875
Iterations		586	894		130	156		
Time		564.0	856.9		135.3	154.8	10.8	128x101
Mean Error		.1755	.0821		.0078	.0066	.0005	
Maximum Error		159.1	65.47		5.065	5.830	.2455	ω=1.910
Iterations	82	478	1000	168	206	252		
Time	83.1	451.3	929.9	168.9	211.6	255.0	9.4	64x201
Mean Error	.7928	.5290	.3923	.0334	.0168	.0075	.0006	
Maximum Error	1542.9	1124.9	814.6	47.81	19.46	8.189	.4928	ω=1.948

TABLE II

Test Solutions of Poisson Equation

eight minutes of computer time, in the performance of 522 iterations, reduced the mean error by only twenty-five percent, and that the errors in the resulting solution are such that it contains no significant figures!

The performance of the Successive Over-relaxation is considerably better, though still inadequate for the long-term integration. The results for the 128 by 101 and the 64 by 201 point arrays indicate that solutions with two significant figures can be obtained in from two to four minutes of computer time.

The rate of convergence which occurs at any point in an iterative solution is proportional to the difference between the current estimate and the solution. As the gap is reduced the rate of approach decreases to some limiting value. It is this asymptotic convergence rate which is the significant factor in the determination of the computational effort required to obtain a solution of specified accuracy (Varga 1962). The three tests with the Successive Over-relaxation in the 64 by 201 point array permit the estimation of two successive convergence rates. The second rate is only very slightly less than the first: their ratio is about .97. These are therefore close to the asymptotic rate. Thus, in this application, the asymptotic rate of convergence is achieved when the mean errors are between one and five percent. If the initial guess is within (say) five percent of the solution, the performance of forty or fifty iterations will reduce the error by only one-half. The final values may have no more significant figures than the first guess.

It is sometimes suggested that, in a numerical integration of the equations for fluid motion, if the time step is kept small, and if the

previous values of the stream function are used as a first guess for the new values, only a few iterations are needed for the solution of the Poisson equation. It is evident that this is not true if the number of mesh points is large. Even if only a very small change in the stream function, or pressure, is required, only a very small fraction of that change will be accomplished with a few iterations. Significant errors will be introduced at each time step. These errors, due to incomplete convergence, unlike round-off errors, are systematic in some sense and unacceptable in a long-term numerical integration.

The errors in the direct solution by the Fourier transform arise from round-off and, in every case tested, were less by at least a factor of ten than the best iterative solution. The computing time was far less, and generally equal to that required for about nine iterations by the Over-relaxation. In the case of the 201 by 64 point array the Transform method produced a solution with errors less, by a factor of ten, than the best iterative solution, and reduced the computing time by a factor of twenty-five.

E. DESCRIPTION OF THE PROGRAM AND COMPUTATIONAL PROCEDURE

A listing of the computer program is reproduced as Appendix B. The names and functions of the various subroutines, the definitions of the parameters which control the operation of the program, as well as the dimensions of the various arrays, are given at the beginning of the calling program. The program is written in standard FORTRAN IV. Calculations were performed on an I.B.M. 360/67 computer at the W. R. Church computer center of the Naval Postgraduate School. Object codes were obtained with the I.B.M. H-level compiler, release 18. Data

storage requirements are the equivalent of 4 M N + 26 N + 13 M single-precision words, where N is the number of mesh points in the Y direction and M the number in the X direction. The program itself requires the equivalent of about 10^4 single-precision words of main storage. The calculations with a 201 by 64 point mesh required 13 seconds of computer time for the performance of a time step. The program incorporates provisions which enable a long integration to be performed in smaller segments. After a specified number of time steps all quantities involved in the integration are written out on a magnetic tape or disc. The integration can be restarted by reading the data from the tape or disc.

The finite-difference mesh is specified by assigning, values to the parameters N, M and ALPHA. The last quantity is the basic period of the motion in units of L, one-half the distance between the parallel walls. It fixes the external streamwise dimension of the finite-difference mesh. In all calculations thus far ALPHA was equal to the wave length of the primary unstable mode.

The time step, δt, is controlled by the parameters FRACEL and XLAMDA. FRACEL is generally the operating parameter. It is equivalent to f in equation (4.43) and expresses the constraint on δt associated with the advective instability of the central time differencing. XLAMDA is an upper limit on the ratio

$$\frac{4\,\delta t}{R\,\delta x\,\delta y}.$$

Fixing this ratio insures that the consistency condition, (4.30), is satisfied. All calculations were performed using a value of 1/4 for XLAMDA.

The parameter KINT specifies the number of time steps between the recalculations of δt. At intervals of KINT time steps: the maximum components of horizontal and vertical velocity are determined; the value of δt is adjusted; values of the vorticity are averaged over the last two steps; values of the stream function recalculated; and finally a single time step is performed by the Modified Euler method.

Subroutines SETUP and START begin the operation. Values of the parameters associated with the finite-difference mesh and various other fixed quantities are established in the former. The initial values of the disturbance stream function and vorticity are established in the latter routine.

The subroutines which perform the subsequent integration and their functions are listed below.

ADVANC----------Time step by central differencing.

STEP----------Time step by Modified Euler differencing.

PRESUR----------Solution of Poisson's equation for the stream function. Calculation of vorticity at the walls.

DHARM----------Calculation of Discrete Fourier transform.

TRISOL----------Solution of tridiagonal system of equations.

TIMER----------Calculation of δt.

Subroutines LOAD and RELOAD perform the read and write operations associated with the interruption and restart of the integration. If the data parameter NPROB is less than zero then the main program calls subroutines SETUP and START for the initiation of a new problem. Upon the completion of the specified number of time steps the values of the various quantities are written on a tape or disc by subroutine RELOAD.

The remaining subroutines are associated with the calculation and printing of various derived quantities. The frequency at which the derived quantities are calculated and printed is controlled by the parameters MSCIP and ISCIP. At intervals of MSCIP time steps the main program calls subroutine PRINT. Subroutines EDDYS, ENERGY, and STRESS are called in sequence. The current values of time, δt, the normalized integrals of kinetic energy in the mean flow, the turbulent motion and of total kinetic energy are calculated and printed. At intervals of ISCIP time steps the main program calls subroutine PLOT. Values of the velocity and vorticity of the mean flow as well as the energy and Reynolds stress associated with the turbulent motion, as functions of Y, are printed and plotted. In addition three calling sequences are started in subroutine PRESUR which involve subroutines SPCTRM, MODES and PLOTSP. Values of the spectral densities of squared vorticity and kinetic energy as functions of Y, as well as the normalized sums over Y of those quantities are calculated, printed and plotted. The functions of Y associated with specified disturbance modes are printed.

F. FINITE DIFFERENCE REPRESENTATION OF THE LINEAR EIGENVALUE PROBLEM

The stream function and vorticity are represented by functions which have the form of equation (3.12), but which are discrete in y. Derivatives with respect to y in equation (3.15) are replaced by the same second-order central differences employed for the non-linear case. The result is an algebraic eigenvalue problem. Equation (3.15) is approximated by equations of the form

$$W\bar{\Phi} = i\beta\bar{\Phi}$$

where, in the general case, W is an (N-2) x (N-2) complex matrix with eigenvalues β and eigenvectors $\bar{\Phi}$. Of particular interest is the effect of N, the number of mesh points between the walls, on the principle eigenvalues of W.

Let

$$\psi = \Phi_J \, e^{-i(\alpha x - \beta t)} + \Phi_J^* \, e^{i(\alpha x - \beta^* t)}$$
$$\Gamma = \theta_J \, e^{-i(\alpha x - \beta t)} + \theta_J^* \, e^{i(\alpha x - \beta^* t)} \qquad (4.57)$$

Equation (3.13) is satisfied if

$$i\alpha R \left[(\beta/\alpha - U_J) \, \theta_J + U'' \Phi \right] - \frac{\theta_{J+1} - (2+\alpha^2 \delta y^2) \, \theta_J + \theta_{J-1}}{\delta y^2} = 0 \qquad (4.58)$$

$$J = 2, 3, \cdots N-1$$

where from equation (3.14)

$$\theta_J = \frac{\Phi_{J+1} - (2 + \alpha^2 \delta y^2) \, \theta_J + \theta_{J-1}}{\delta y^2} \qquad J = 2, 3, \cdots N-1 \qquad (4.59)$$

The boundary conditions require that

$$\Phi_1 = \Phi_N = 0$$

and that, in accordance with the previous formulation of the vorticity boundary conditions,

$$\theta_N = \frac{P_1 \, \Phi_{N-1} + P_2 \, \Phi_{N-2} + P_3 \, \Phi_{N-3} + P_4 \, \Phi_{N-4}}{\delta y^2}$$
$$\theta_1 = \frac{P_1 \, \Phi_2 + P_2 \, \Phi_3 + P_3 \, \Phi_4 + P_4 \, \Phi_5}{\delta y^2} \qquad (4.60)$$

where for the various approximation for the boundary conditions, equations (4.12), (4.13), (4.14) and (4.15), P_K, K = 1, 2, 3, 4, are given by

TABLE III

COEFFICIENTS FOR VORTICITY BOUNDARY CONDITIONS

	P_1	P_2	P_3	P_4
first order	2	0	0	0
second order	4	-1/2	0	0
third order	6	-3/2	2/9	0
fourth order	122/9	-3	.8/9	-1/8

Equations (4.58) and (4.59) can be written

$$i\beta\theta_J = i\alpha\left[U_J\,\theta_J - U''\,\Phi_J\right] + \frac{\theta_{J+1} + \omega\theta_J + \theta_{J-1}}{R\delta y^2} \tag{4.61}$$

$$\theta_J = \frac{\Phi_{J+1} + \omega\Phi_J + \Phi_{J-1}}{\delta y^2} \tag{4.62}$$

where

$$\omega = -\,(2 + \alpha^2\,\delta y^2) \tag{4.63}$$

We define the following Matrices and vectors.

$$B = \begin{bmatrix} \omega & 1 & & & & \\ 1 & \omega & 1 & & & \\ & 1 & \omega & 1 & & \\ & & & \cdots & & \\ & & & 1 & \omega & 1 \\ & & & & 1 & \omega \end{bmatrix} \qquad P = \begin{bmatrix} P_1 & P_2 & P_3 & P_4 & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & P_4 & P_3 & P_2 & P_1 \end{bmatrix} \tag{4.64}$$

$$\bar{\theta} = \begin{Bmatrix} \theta_2 \\ \theta_3 \\ \cdot \\ \cdot \\ \cdot \\ \theta_{N-2} \\ \theta_{N-1} \end{Bmatrix} \qquad \bar{\Phi} = \begin{Bmatrix} \Phi_2 \\ \Phi_3 \\ \cdot \\ \cdot \\ \cdot \\ \Phi_{N-2} \\ \Phi_{N-1} \end{Bmatrix} \qquad \bar{\gamma} = \begin{Bmatrix} \theta_1 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ \theta_N \end{Bmatrix} \tag{4.65}$$

And adopt the notation for a diagonal matrix with diagonal elements

$$\eta_J \qquad J = 2,3,\cdots N-1$$

$$\Lambda(\eta_J) = \begin{bmatrix} \eta_2 & & & & & & \\ & \eta_3 & & & & & \\ & & \eta_4 & & & & \\ & & & \cdot & & & \\ & & & & \cdot & & \\ & & & & & \eta_{N-2} & \\ & & & & & & \eta_{N-1} \end{bmatrix} \tag{4.66}$$

Equations (4.61) and (4.62) become

$$i\beta\bar{\theta} = \Lambda(i\alpha U_J)\ \bar{\theta} - \Lambda(i\alpha U'')\ \bar{\Phi} + \frac{1}{R\delta y^2}\ [B\ \bar{\theta} + \bar{\gamma}] \tag{4.67}$$

$$\bar{\theta} = \frac{1}{\delta y^2}\ B\bar{\Phi} \tag{4.68}$$

The vorticity boundary conditions, equations (4.60), become

$$\bar{\gamma} = \frac{1}{\delta y^2}\ P\bar{\Phi} \tag{4.69}$$

The boundary conditions, $\Phi_1 = \Phi_N = 0$ are satisfied by equations (4.67) and (4.68). Substituting equations (4.68) and (4.69) in (4.67) we obtain

$$i\beta\, B\bar{\Phi} = \left\{ i\left[\Lambda(\alpha U_J)B - \Lambda(\alpha U''\delta y^2)\right] + \frac{1}{R\delta y^2}\left[B^2 + P\right]\right\}\bar{\Phi}$$

Multiplying through by B^{-1} we obtain

$$i\beta\Phi = \left\{\frac{1}{R\delta y^2}\left[B + B^{-1}P\right] + iB^{-1}\left[\Lambda(\alpha U_J)B - \Lambda(\alpha U''\delta y^2)\right]\right\}\bar{\Phi} \qquad (4.70)$$

Thus

$$i\beta\bar{\Phi} = W\bar{\Phi} \qquad (4.71)$$

Where

$$W = \frac{1}{R\delta y^2}\left[B + B^{-1}P\right] + iB^{-1}\left[\Lambda(\alpha U_J)B - \Lambda(\alpha U''\delta y^2)\right] \qquad (4.72)$$

For the linear equations the problem is simply the determination of the eigenvalues and eigenvectors of a complex matrix. Because of symmetry of the laminar velocity distribution the symmetric and anti-symmetric solutions of both the differential and algebraic equations are uncoupled. It is useful to take advantage of this fact to reduce the order of the matrix W. The well-known unstable solutions for the Plane Poiseuille flow are all symmetric. It appears that all the anti-symmetric solutions are stable (Grosch 1968). In the present undertaking only symmetric modes are considered.

Thus we consider only the region

$$-1 \leq y \leq 0$$

and add the boundary conditions at the centerline ($y = 0$, $J = \frac{N+1}{2}$)

$$\Phi' = \Phi''' = \cdots = 0$$

In terms of the finite differences this is equivalent to

$$\Phi_{\frac{N-1}{2}} = \Phi_{\frac{N+3}{2}} \qquad \theta_{\frac{N-1}{2}} = \theta_{\frac{N+3}{2}}$$

The matrix formulation remains the same except that the order of the various matrices and vectors is reduced from N-2 to $\frac{N-1}{2}$, and equations (4.64) and (4.65) for B, P, $\bar{\Phi}$, $\bar{\theta}$ and $\bar{\gamma}$ become

$$B = \begin{bmatrix} \omega & 1 & & & & \\ 1 & \omega & 1 & & & \\ & 1 & \omega & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & & 1 & \omega & 1 \\ & & & & & 2 & \omega \end{bmatrix} \qquad P = \begin{bmatrix} P_1 & P_2 & P_3 & P_4 \end{bmatrix} \tag{4.73}$$

$$\bar{\theta} = \begin{Bmatrix} \theta_2 \\ \theta_3 \\ \cdot \\ \cdot \\ \cdot \\ \theta_{\frac{N-1}{2}} \\ \theta_{\frac{N+1}{2}} \end{Bmatrix} \qquad \bar{\Phi} = \begin{Bmatrix} \Phi_2 \\ \Phi_3 \\ \cdot \\ \cdot \\ \cdot \\ \Phi_{\frac{N-1}{2}} \\ \Phi_{\frac{N+1}{2}} \end{Bmatrix} \qquad \bar{\gamma} = \begin{Bmatrix} \theta_1 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \end{Bmatrix} \tag{4.74}$$

The eigenvalues and eigenvectors of equation (4.71) were computed numerically by means of the Q R algorithm (Wilkinson 1965, Parlette 1966). This algorithm, for the solution of an algebraic eigenvalue problem, was developed by Francis (1962). It involves a sequence of unitary transformations which yield an upper canonical form of the eigenvalue problem. The inverse of the matrix B in equation (4.70) is obtained by standard Gauss-Elimination. All calculations were performed in double precision (sixteen decimal digits) on an I.B.M. 360/67 computer. Standard I.B.M. scientific library subroutines, with some

minor modifications, were employed for the calculation of the inverse of B and the eigenvalues and eigenvectors of W. A listing of the FORTRAN program is reproduced as Appendix C.

V. RESULTS AND DISCUSSION

A. LINEAR CALCULATIONS

Symmetric solutions of the algebraic eigenvalue problem, equation (4.71), were obtained at regularly spaced points in a portion of the wave number-Reynolds number domain for various values of N ranging from 41 to 201. The investigation was confined to the vicinity of the unstable region at Reynolds numbers out to 25000. The calculations yield, in each case, N/2 symmetric eigenfunctions and the N/2 associated eigenvalues. A summary of selected results is depicted in Figs. 4-11.

Preliminary calculations which employed the various order formulations of the vorticity boundary conditions, Table III, indicated substantial differences in the solutions with the first and second order formulations but only slight differences between solutions based on second and higher order formulations. Consequently the consistent second-order formulation was adopted for the subsequent linear calculations, as well as for the numerical integration.

Calculations with N = 41 revealed no indication of linear instability, ($\beta_I < 0$), at any point in the region investigated. Calculations were performed for wave numbers from .66 to .94 in increments of .02 for a regular sequence of Reynolds numbers out to 25000. In every case the calculated values of β_I reached a local minimum at a Reynolds number less than 20000. Consequently it is believed that a second-order finite-difference representation of the equations of motion with $N \leq 41$, ($\delta y \geq .05$), completely obscures the phenomenon of linear instability. For $N > 41$, points on the neutral curve were obtained by interpolation.

Neutral curves, determined in this manner, for N = 61, 101 and 201, are shown in Fig. 4, along with the neutral curve established by Grosch. The agreement with established results is poor for N = 61, fair for N = 101 and for N = 201 reasonably good. Although it is not shown, the neutral curve for N = 161 differs only slightly from that for N = 201 and only on the upper branch. The accuracy of the representation, for a given value of N, is greatest in the vicinity of the lower branch of the neutral curve. This is illustrated in Fig. 5, where the calculated growth rates, β_I, are plotted versus the wave number, α, for a Reynolds number of 15000. Solutions for the eigenvalues of the least stable mode, in a sequence of calculations with increasing N converge most rapidly for the smaller values of α. The same kind of behavior characterizes the convergence of the asymptotic expansions for the solution to the linear differential equation (Potter 1966).

The sets of calculated eigenvalues for the cases $\alpha = 1.0$, $R = 7333.3$, 11333.3 and 25000 are plotted in Figs. 6, 7, and 8. The orderly arrangements of the eigenvalues in the complex plane are quite striking. None of the eigenvalues has a phase velocity, β_R, greater than 1.5, the maximum velocity of the laminar flow, and, at the lower Reynolds numbers, the majority have phase speeds equal to the mean laminar velocity. The effect of increased Reynolds number is evidently the greatest on the most stable modes. As the Reynolds number is increased the eigenvalues approach the neutral line ($\beta_I = 0$). As $R \to \infty$, the matrix W in equation (4.71) becomes Hermitian. In the limit all the solutions are neutrally stable.

A measure of the accuracy of the calculations can be obtained by comparison with selected results of Grosch (1968) and Thomas (1953). Calculated growth rates in the unstable region for N = 101, 161 and 201, as well as values calculated by Grosch are plotted in Figs. 9 and 10, for Reynolds numbers 7333.3 and 11333.3 respectively. The very slow convergence of the second order finite-difference scheme in the vicinity of the upper branch of the neutral curve is evident.

In terms of the reference units employed here, the primary eigenvalues calculated by Thomas were, for $\alpha = 1.0$, R = 1066.6 and R = 6666.7 respectively

$$\beta = .4847 + i \quad .0393$$

$$\beta = .3563 - i \quad .0055$$

For these cases the present calculations with N = 201 yield

$$\beta = .4891 + i \quad .0393$$

$$\beta = .3593 - i \quad .0041$$

A comparison of the calculated eigenvectors for the case R = 6666.7, $\alpha = 1.0$ is given in Fig. 11. Results for the real part of the eigenvector are identical to four significant digits. There are however small differences in the imaginary parts. Thomas employed a transformation which made his finite-difference approximation basically of fourth order. It is not surprising that a second-order scheme might produce slightly different results for a function with large derivatives and curvature. However test calculations, with a linearized form of the numerical integration, which employed the calculated linear solutions as initial conditions, suggest that most of the differences are attributable to slight numerical errors in the present solutions.

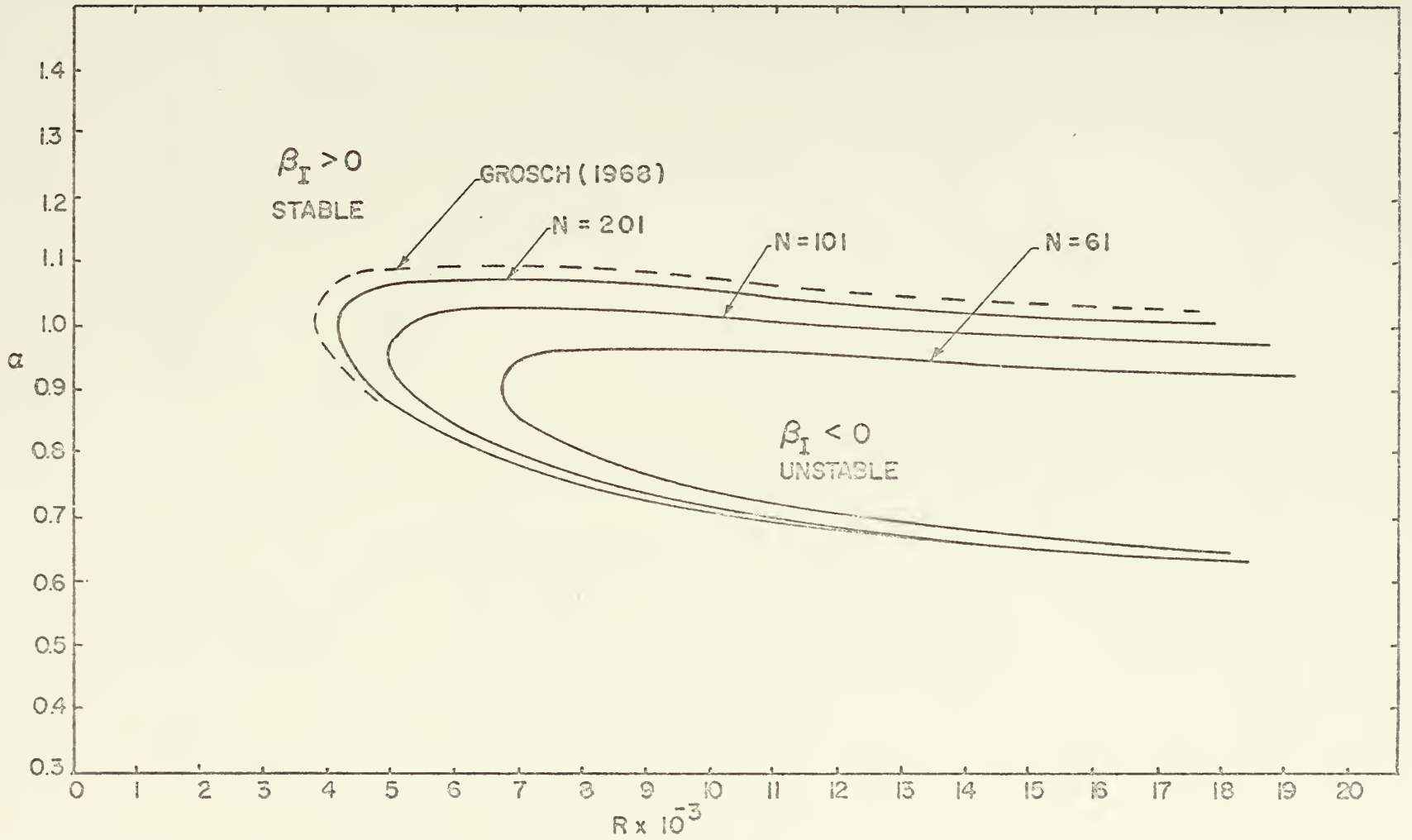


FIGURE 4 EFFECT OF FINITE DIFFERENCE MESH SIZE ON CURVE OF NEUTRAL STABILITY FOR PLANE POISEUILLE FLOW.

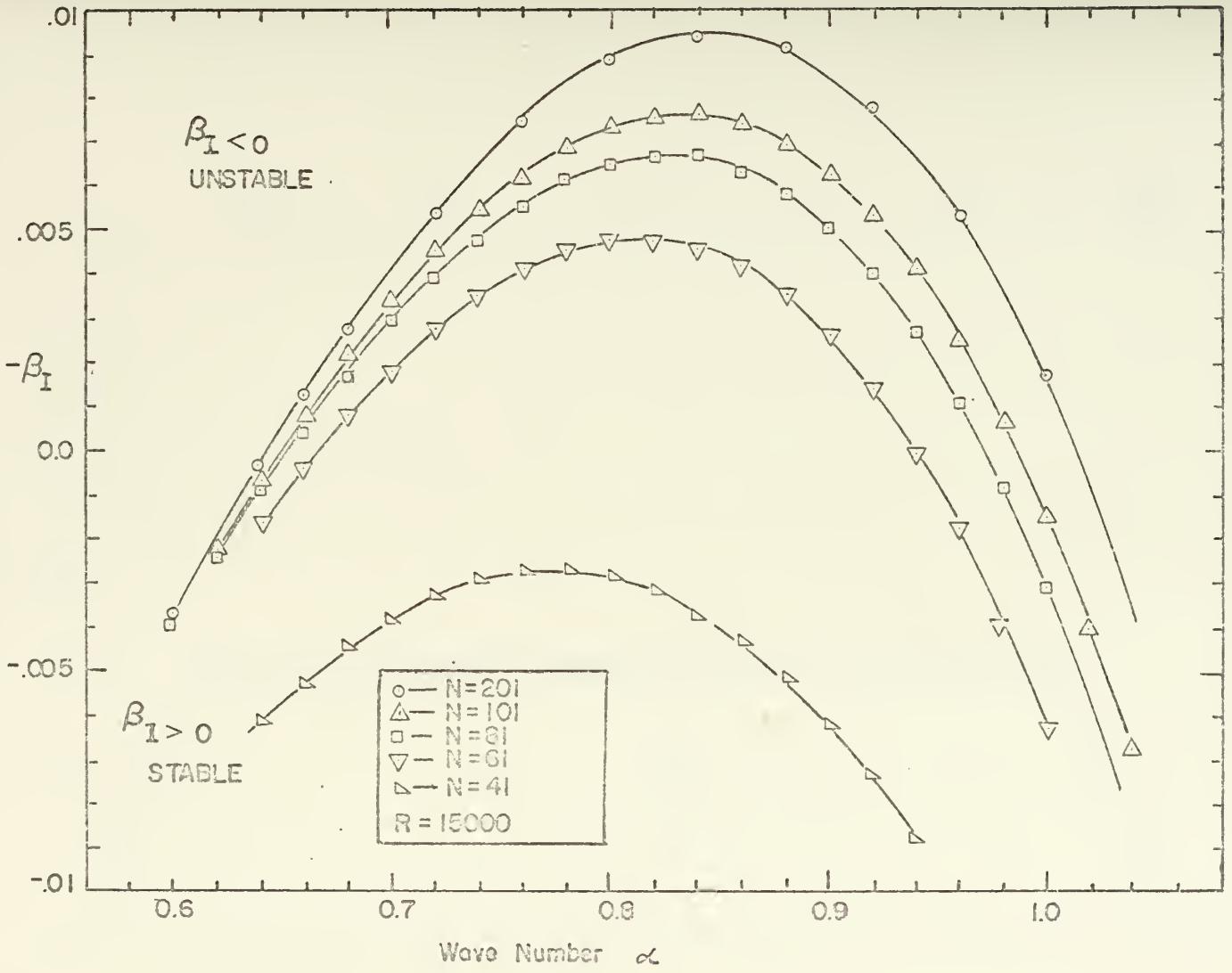


FIGURE 5 EFFECT OF N ON CALCULATED GROWTH RATE

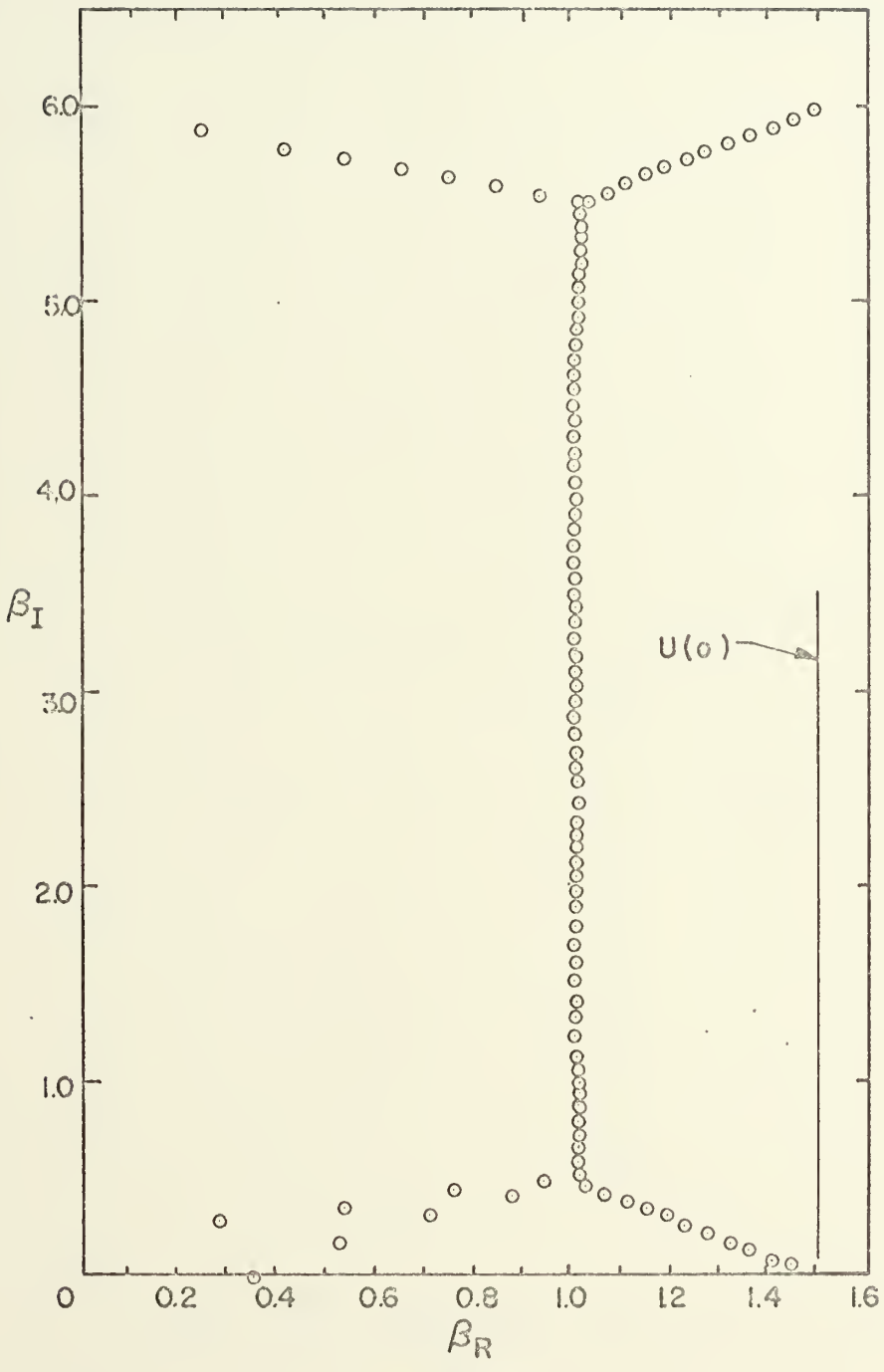


FIGURE 6

CALCULATED EIGENVALUES - SYMMETRIC MODES

R = 6666.7 α = 1.0 N = 201

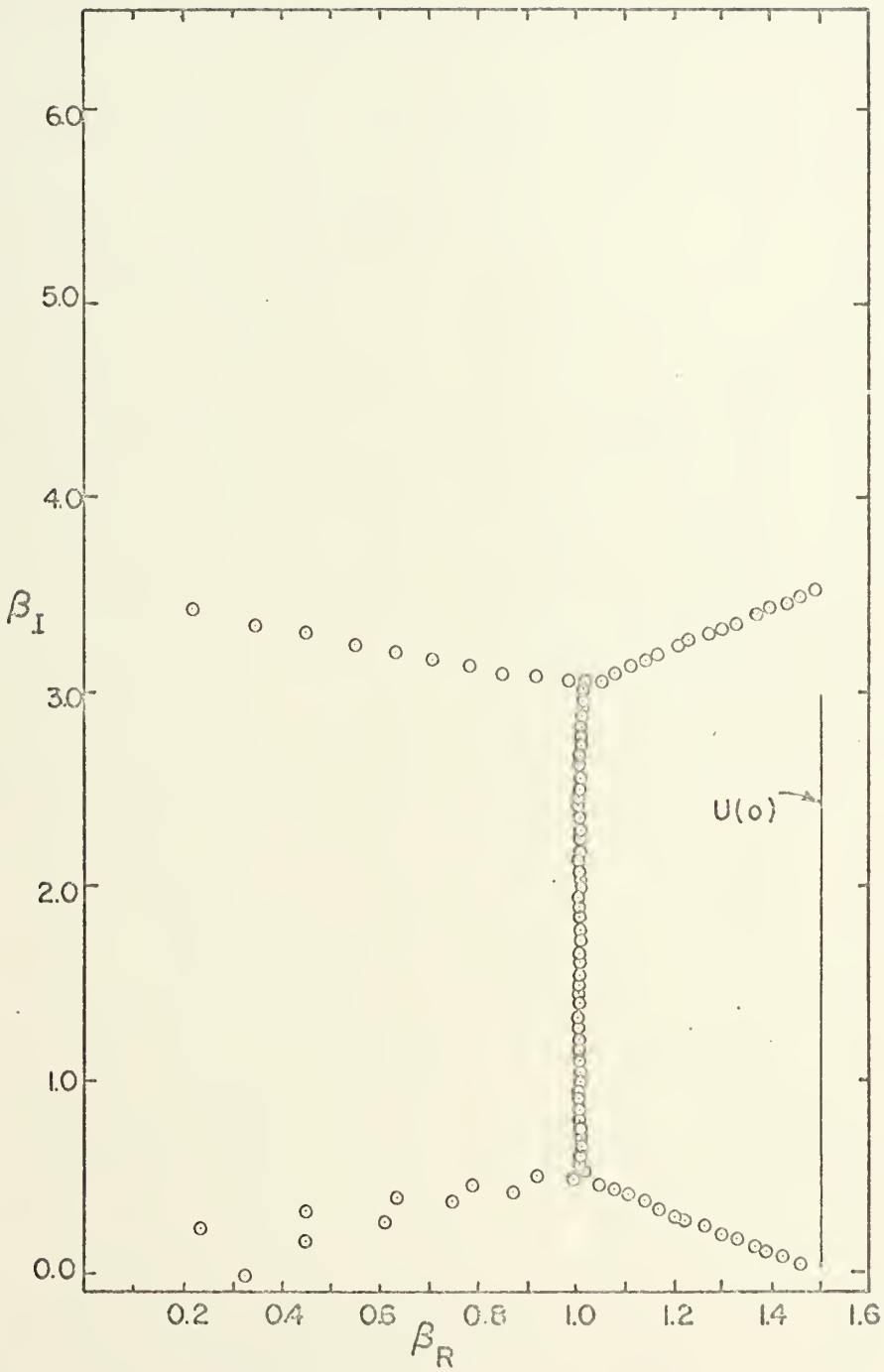


FIGURE 7

CALCULATED EIGENVALUES - SYMMETRIC MODES

R = 11333.4 α = 1.0 N = 201

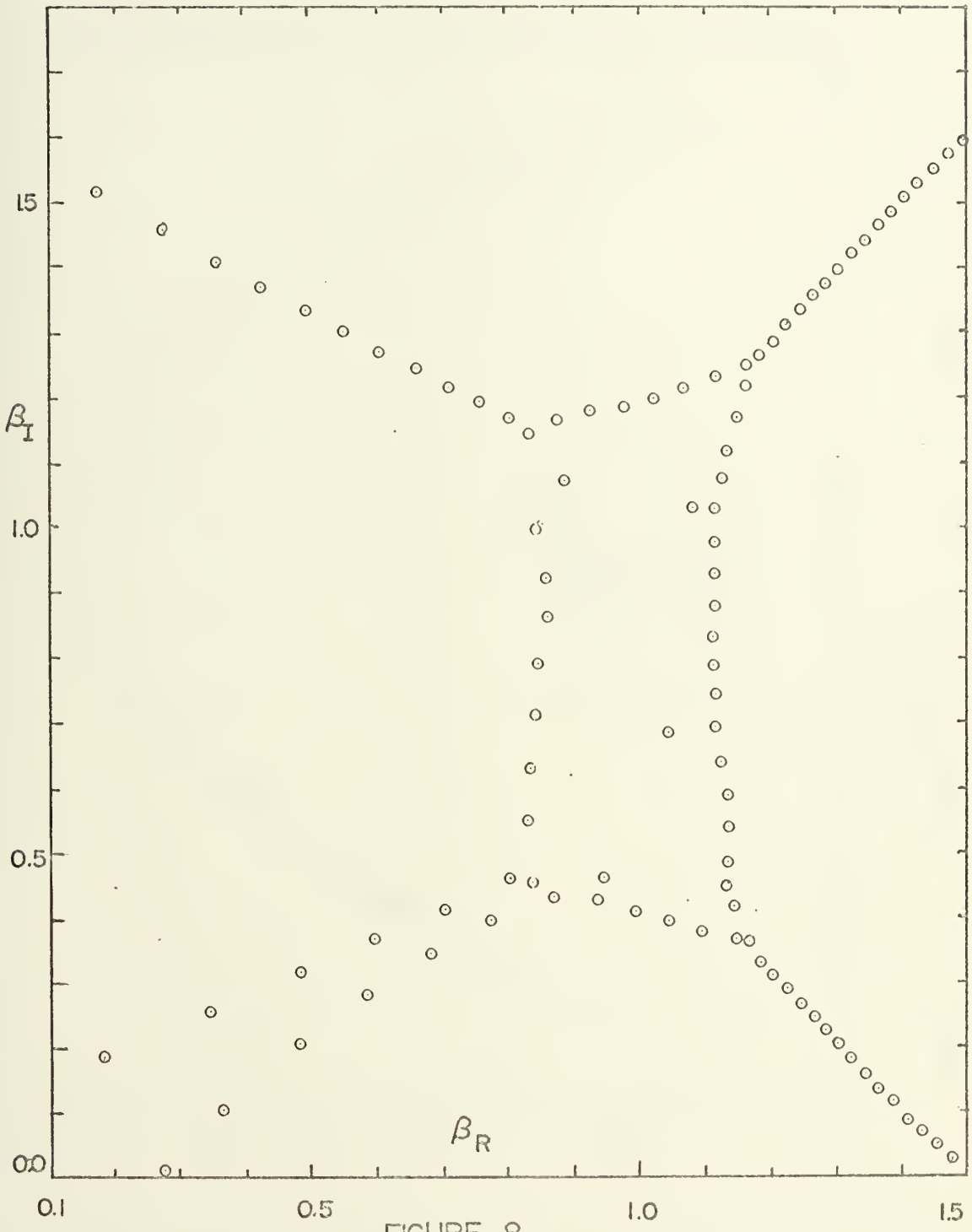


FIGURE 8

CALCULATED EIGENVALUES — SYMMETRIC MODES

R = 25,000 α = 1.0 N = 201

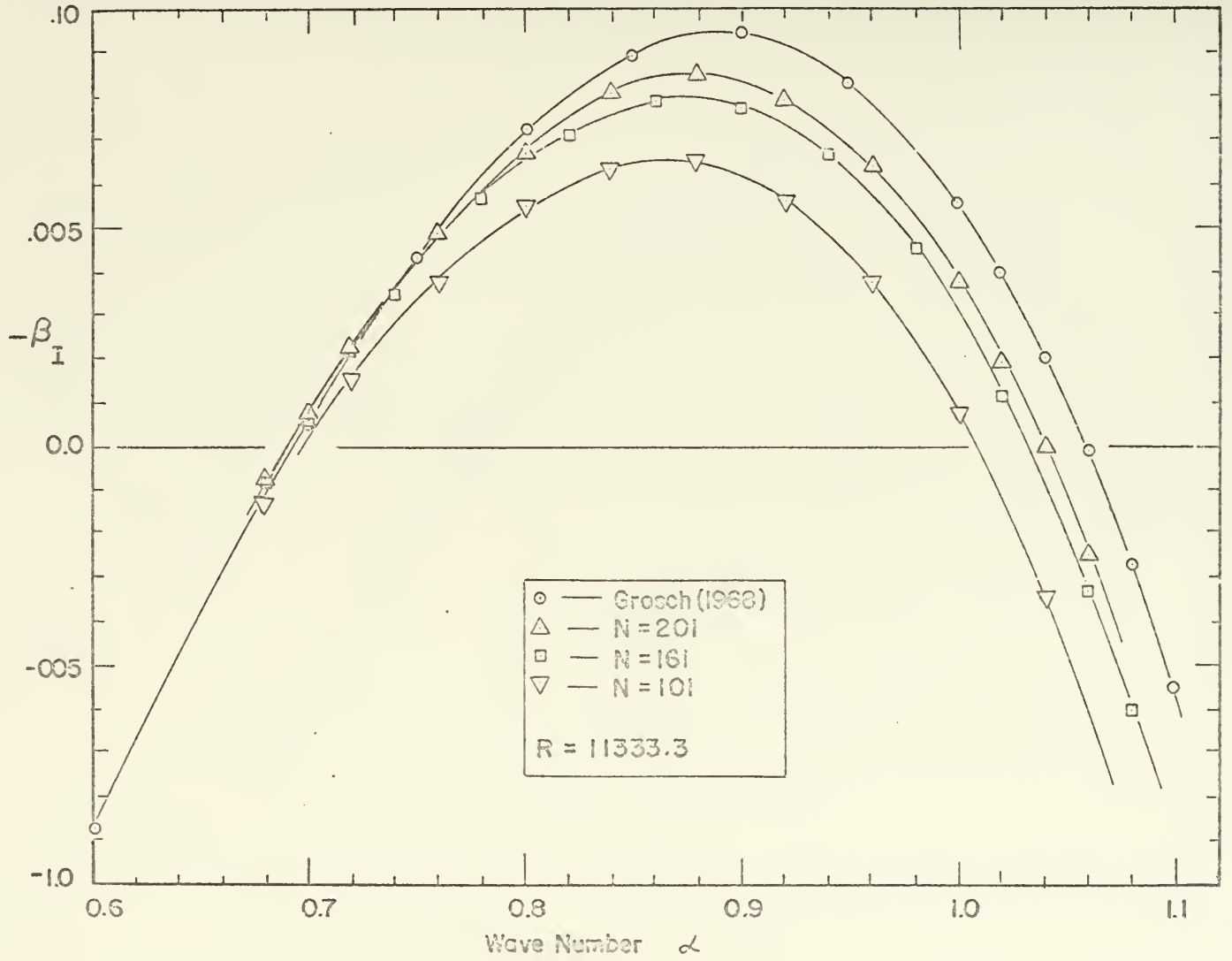


FIGURE 9 COMPARISON OF CALCULATED GROWTH RATES.

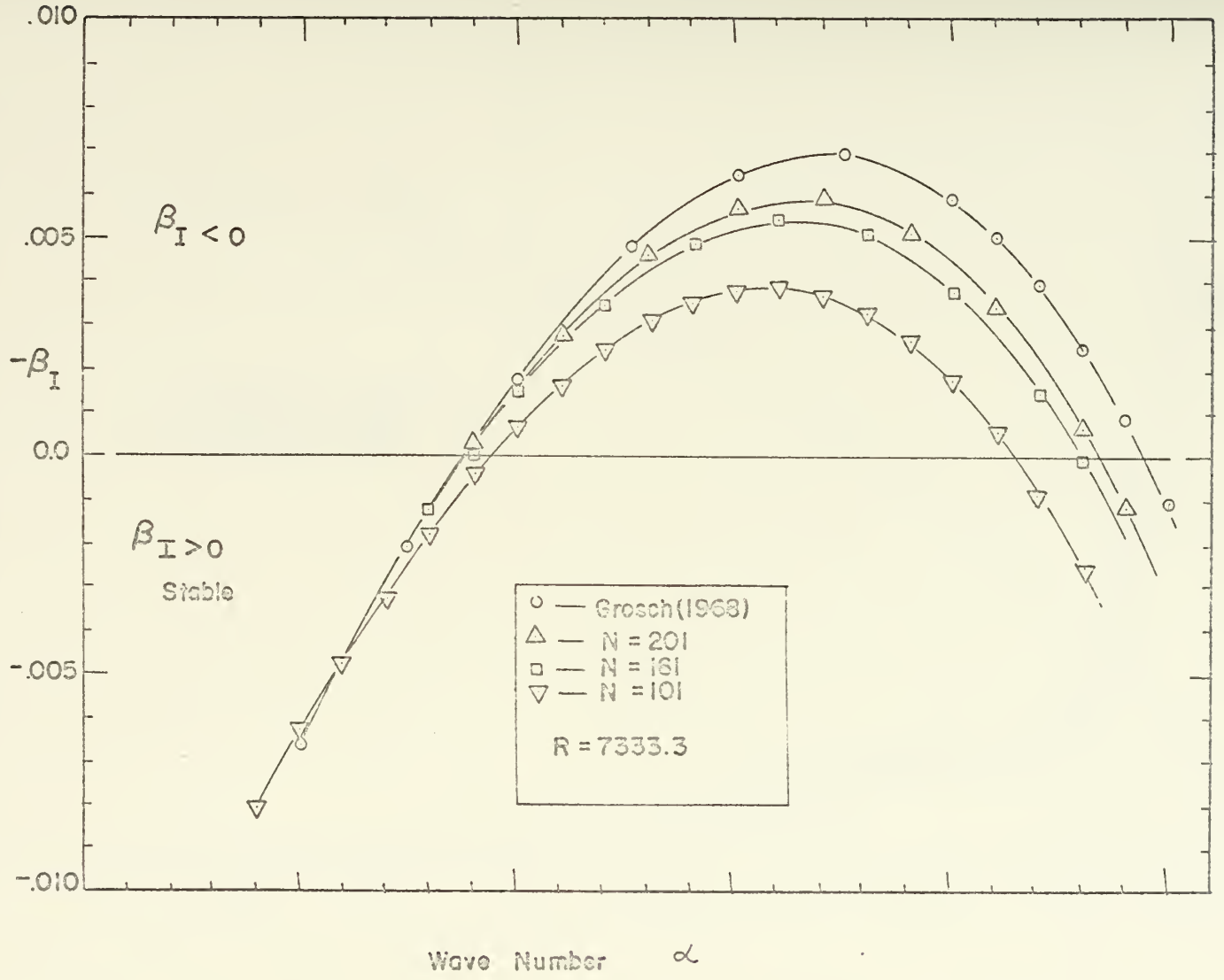


FIGURE 10 COMPARISON OF CALCULATED GROWTH RATES.

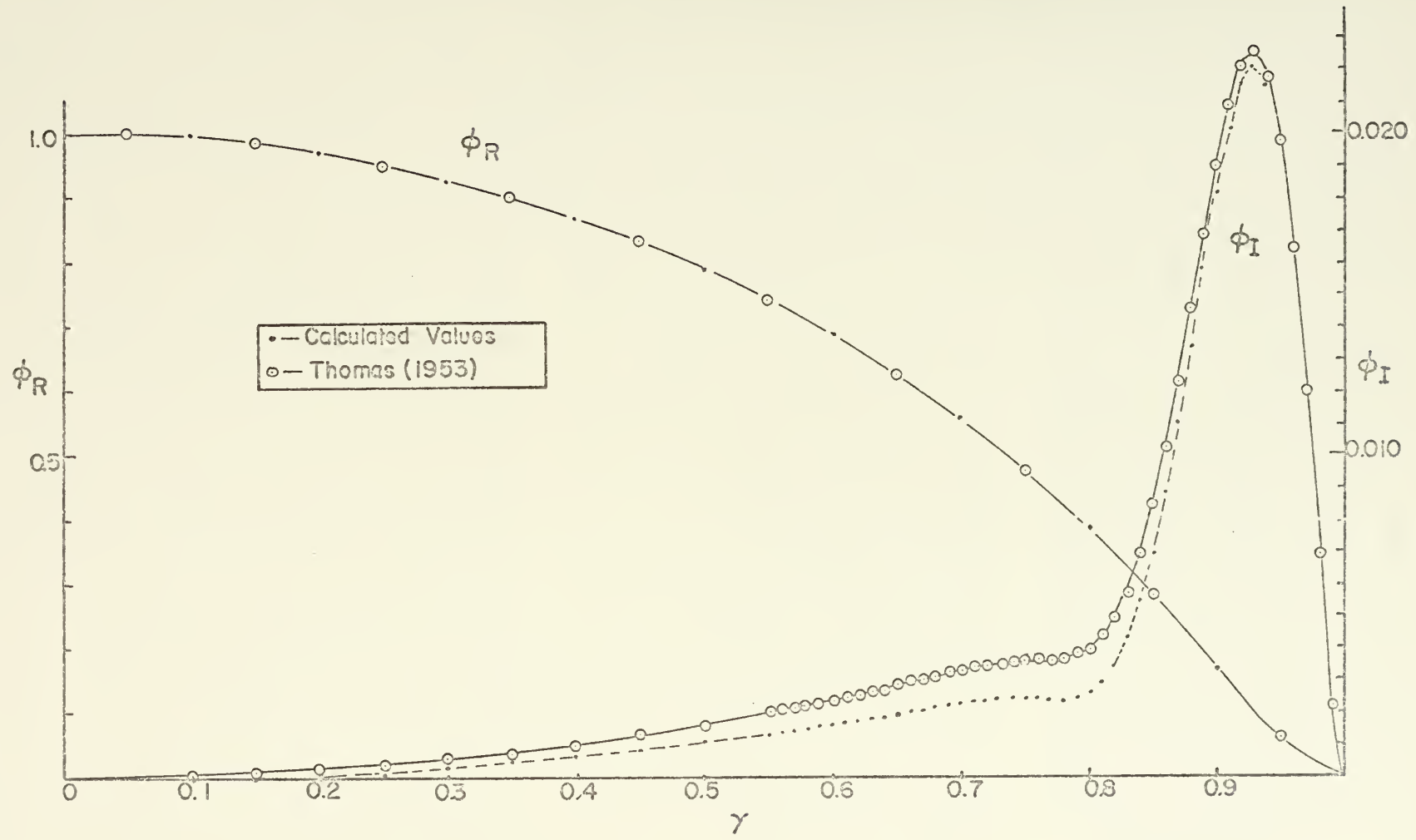


FIGURE II COMPARISON OF CALCULATED EIGENVECTORS $R = 6666.7$ $\alpha = 1.0$

The errors associated with a particular eigen-solution by the Q. R. algorithm are dependent, among other things, on the ratio of the modulus of the associated eigenvalue to the largest such modulus. Decreased values of that ratio imply increased errors in the solution. In the present application the eigenvalue associated with the least stable mode has the smallest modulus, and it is reasonable to expect that there might be some accumulation of numerical error in the solution for that mode.

In retrospect it seems that, for the linear solutions, an iterative method which focuses on a single mode, such as the approach introduced by Nachtscheim (1964), might have been more appropriate for the limited and definite purposes of the present undertaking. However in any application in which there is interest in the higher modes the matrix approach has definite advantages.

B. NUMERICAL INTEGRATIONS

1. Preliminary

The linear calculations, in addition to providing a measure of the resolution required to represent adequately the two dimensional motions, provide a rational means of establishing initial conditions for the numerical integration. In particular they provide a unique means of testing the stability (in the linear range) and accuracy of the finite-difference formulation and solution. The calculation of the non-linear terms in the finite-difference solution can be bypassed by minor changes in the FORTRAN code. If this is done, the scheme should be capable of reproducing the calculated linear eigensolutions. Specifically, if an unstable mode is employed to establish the initial conditions, the numerical integration should produce the predicted

growth rate and phase speed. The variation of the calculated turbulent kinetic energy with time should be, from equation (3.18), $e^{-2\beta_I t}$. Calculations of the spectral density of turbulent kinetic energy and squared vorticity should reveal no spurious excitation in other represented wave lengths. Finally, perhaps the most stringent requirement is that the performance of a large number of time steps should not produce any significant distortion of the normalized eigenvector.

If the scheme can be demonstrated to be stable and accurate, at least in the linear features, then, since the virtues of the particular non-linear representation have been fairly well established elsewhere, calculations can be performed in the non-linear range with confidence in the accuracy of the results.

The resolution requirements indicated by the linear calculations are very great: about 201 mesh points are required in the Y direction. The requirements for resolution in the X direction are determined by the extent to which higher harmonics, or smaller scales of motion, are excited by the non-linear terms. M points in the X direction represent M/2 Fourier modes. As a result of the well-known constraints on energy cascades in a two-dimensional motion the extent of the excitation of the higher harmonics should not be great. Kraichnan (1967) has predicted a -3 power law for the spectral density of turbulent kinetic energy in the smaller scales. If this is the case, then the energy in wave number 10, for example, should be less by three decades than that in the primary mode of the disturbance. If the basic period of the motion, the external downstream dimension of the finite-difference mesh, is chosen to be equal to the period of the unstable mode, a mesh with M = 64, which represents 32 Fourier modes, should be adequate.

2. Results

A tabulation of information which serves to identify the various integrations or runs which will be referred to in this section is given in Table IV.

TABLE IV

SUMMARY OF NUMERICAL INTEGRATIONS

Run	Initial Conditions	2σ	E(o)	Number of Time Steps	Time	Computer Time (hrs)
1	B*	.3	$.4\times10^{-1}$	1000	11.7	4.8
2	A*	$.1\sqrt{5}$	$.25\times10^{-1}$	1550	20.0	6.1
3	A	.01	$.5\times10^{-4}$	800	8.3	3.8
4	A	$.01\sqrt{10}$	$.5\times10^{-3}$	800	8.3	3.8
5	A	.1	$.5\times10^{-2}$	800	8.2	3.8
6	A	$.1\sqrt{10}$	$.5\times10^{-1}$	800	7.6	3.8
7	A	1.0	.5	3700	11.0	17.5
8	A	$.1\sqrt{5}$	$.25\times10^{-1}$	5900	61.2	28.0
9	A	$.1\sqrt{2}$	$.25\times10^{-2}$	2700	52.5	12.0

Initial conditions:

$$\psi^{o}_{K,J} = \sigma \left\{ \Phi_J \, e^{i\alpha x_K} + \Phi^{*}_J \, e^{-i\alpha x_K} \right\}$$

A - R = 6666.7, α = 1.0

Φ_J from Thomas (1953)

B - R = 15000, α = .86

Φ_J from present calculations

*non-linear terms suppressed.

As indicated in the table, runs one and two were performed with the non-linear terms suppressed. The remaining runs included the non-linear interactions and were at a Reynolds number of 6666.7. The linear solution calculated by Thomas was employed as an initial condition. They differ in the initial amplitude, σ.

Preliminary test calculations, with the non-linear terms suppressed, which employed, as initial conditions, the primary eigenfunctions calculated here indicated not only the accuracy of the integration scheme, but also a slight inaccuracy in the calculated eigenvectors. Run number one is typical of these calculations, most of which were performed in fairly coarse arrays. The logarithm of the ratio of the turbulent energy to its initial value, in that run, is plotted as a function of time, along with predicted rate in Fig. 12. During the course of an initial transient which persists for about six units of time, the growth rate increases and then decreases to about the predicted value. Although the errors in the growth rate, during that period, are large the asymptotic value is very good. Other calculations with $N = 41$ and a decaying disturbance exhibited the same behavior. In both cases the predicted rate was achieved and persisted after the transient. Reference to equation (3.19) suggests the cause of the transient, the presence of small errors in the calculated eigenvectors. In effect, small contributions from higher modes are present in the linear solution for the least stable mode. Figures 6 and 7 illustrate the fact that the stability of the great majority of the higher linear modes far exceeds the instability of the single unstable mode; there are wide variations in the exponential rates of growth or decay. After several units of time the energy associated with the

errors is washed away. The integration routine, if it is accurate, acts as a filter and tends to suppress contributions from all but the least stable mode when the non-linear terms are omitted.

A subsequent run, (run number two), at a Reynolds number of 6666.7 which employed the eigenvector calculated by Thomas as initial conditions tends to confirm this suggestion. For this run, the logarithm of the energy ratio, along with both the growth rates predicted in the present calculations and that of Thomas are plotted in Fig. 13. There is a very slow decrease in the growth rate out to about ten units of time. After that point the variation is linear and very close to the growth rate predicted by the second-order calculations. In effect, the eigenvector calculated by Thomas in a basically fourth-order formulation is a better approximation of the second order solution than those obtained in the present work. After 1000 time steps the vector was renormalized and printed, in order to obtain a measure of the distortion. The variations, where they occur, are plotted in Fig. 14. The maximum distortion in amplitude was .035% and in phase 1.5%.

Calculated spectral densities of kinetic energy and squared vorticity revealed no indication of spurious excitation of other spatial wave lengths in any of the linear runs.

Thus it may be concluded that the integration scheme is stable and highly accurate.

Although the performance of the integration routine in the linear tests was most satisfying, the inaccuracies in the calculated eigenvectors and the resulting transient in the linear integration inhibit a most useful application of the program. That is, to the

determination of critical amplitudes at sub-critical values of the Reynolds number. For this application the representation of the linear behavior in the early stages should be more accurate than that which results from the present calculation. However, more accurate methods, for the determination of the solutions of the algebraic equations in a consistent second-order model of the linear problem, are available. In particular the refinements of the iterative approach of Nachtsheim (1964), which were developed and employed by Reynolds are most promising.

In the subsequent non-linear calculations at R = 6666.7 and $\alpha = 1.0$, the linear solution, calculated by Thomas (1953), was employed for the initial condition. The amplitude factor, σ, is based on the normalization of the linear solution given by Thomas. The objectives of the non-linear calculations were, to determine the effect of various amplitudes on the initial development of the disturbance, and, if possible, to achieve an equilibrium state for the two-dimensional turbulence.

In runs three through six the initial amplitudes were such that the initial kinetic energy of the disturbance increased, by decades, from $.5 \times 10^{-4}$ to $.5 \times 10^{-1}$. For reference, the normalized sum of kinetic energy in the laminar motion is .6 in the present units. The growth of turbulent energy for these calculations, as well as for run number nine, are shown in Fig. 15. For run number three, no significant departures from the linear behavior were observed until after five units of time. This provides some measure of the amplitude at which the non-linear interactions become significant. The limit in the growth of the turbulent energy in run number six suggests the existence of an

equilibrium state, in terms of kinetic energy, in the range $.005 < \hat{E} < .05$.

In order to verify the limiting behavior of the turbulence a long calculation was performed at an even higher energy level. This was run number five. In this case the initial kinetic energy of the disturbance was of the same order as the energy in the laminar flow. The subsequent variations in the kinetic energies of the mean flow, the turbulent motion, and of the energy in the primary mode of the disturbance, as well as the total kinetic energy in the domain are shown in Fig. 16. All quantities are normalized with respect to the initial value of the total energy. Points are plotted at intervals of 100 time steps. It is the variation of the total energy which most clearly indicates the trend. It decayed monotonically, and by the end of the run about 13% of the kinetic energy in the system had been dissipated. The early behavior of the turbulence is dominated by large exchanges of energy between it and the mean flow.

Run number eight, in which the initial energy of the disturbance was .025, produced an approximate equilibrium condition. The variation of the various energy terms is shown in Fig. 17. After a slight decrease, the total energy remained virtually constant for the last 25 units of time. Again, there is a periodic exchange of energy between the mean flow and the disturbance. However in this case the period of the oscillation was more than five times as great as in the previous case, and, in the course of the development, the amplitude decreased markedly. A comparison of the variations of the total kinetic energy in the two-dimensional turbulence to the energy in the primary mode suggests that the oscillations are associated with transients in the development of

the higher wave numbers. After 47 units of time the amplitude of the oscillation was much reduced, and the difference between the two kinetic energy terms, which is the total kinetic energy in the higher wave numbers, remained nearly constant. Mean velocity profiles at extreme points in the early variations of the disturbances are given in Fig. 19. Plots of the mean velocity, mean vorticity and turbulent kinetic energy near the final state are given in Fig. 20. The mean velocity profile is only slightly distorted although near the walls there are significant changes in its curvature. Calculated spectral densities of the turbulent kinetic energy are consistent with the -3 power law proposed by Kraichnan. Values of the spectral density of kinetic energy at $y = \pm .85$ are given in Fig. 21, mean values over y, in Fig. 22.

The function of y which describes the primary mode remains symmetric. In addition, it turns out that modes three and five are also symmetric while modes two and four are anti-symmetric. Although the remaining mode shapes were not determined in the calculations, it appears likely that the odd-numbered modes are symmetric and the even-numbered modes anti-symmetric. Mode shapes, in terms of amplitude and phase, for modes one through four are given in Figs. 24 through 27. The initial shape for mode one is given in Fig. 23. It is interesting to note that, although the symmetry and anti-symmetry of the odd and even-numbered modes does not necessarily follow from equations (3.4), it does follow from the equations treated by Watson (1960) and Reynolds (1967), in which certain of the non-linear terms are ignored.

The last calculation, run number nine, was started with a disturbance energy, less by a factor of ten, than that of run eight.

In this case the variations of the various energy terms were nearly monotonic. After 50 units of time the disturbance energy had increased by more than a factor of three. It appears likely that if this calculation were continued the equilibrium energy level would have been attained without the large oscillations which characterized the initial behavior of the previous run.

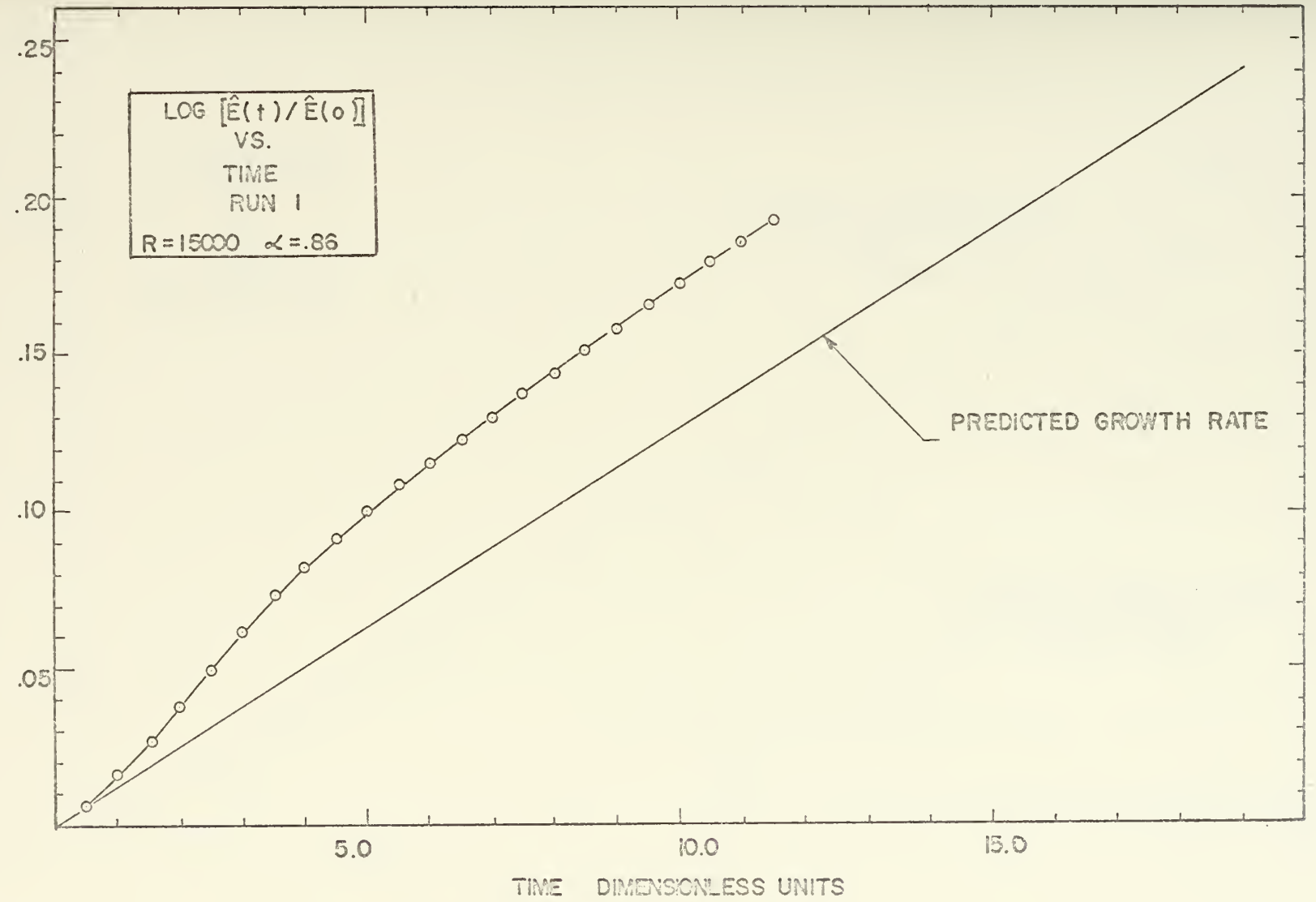


FIGURE 12 TEST FOR LINEAR SOLUTION.

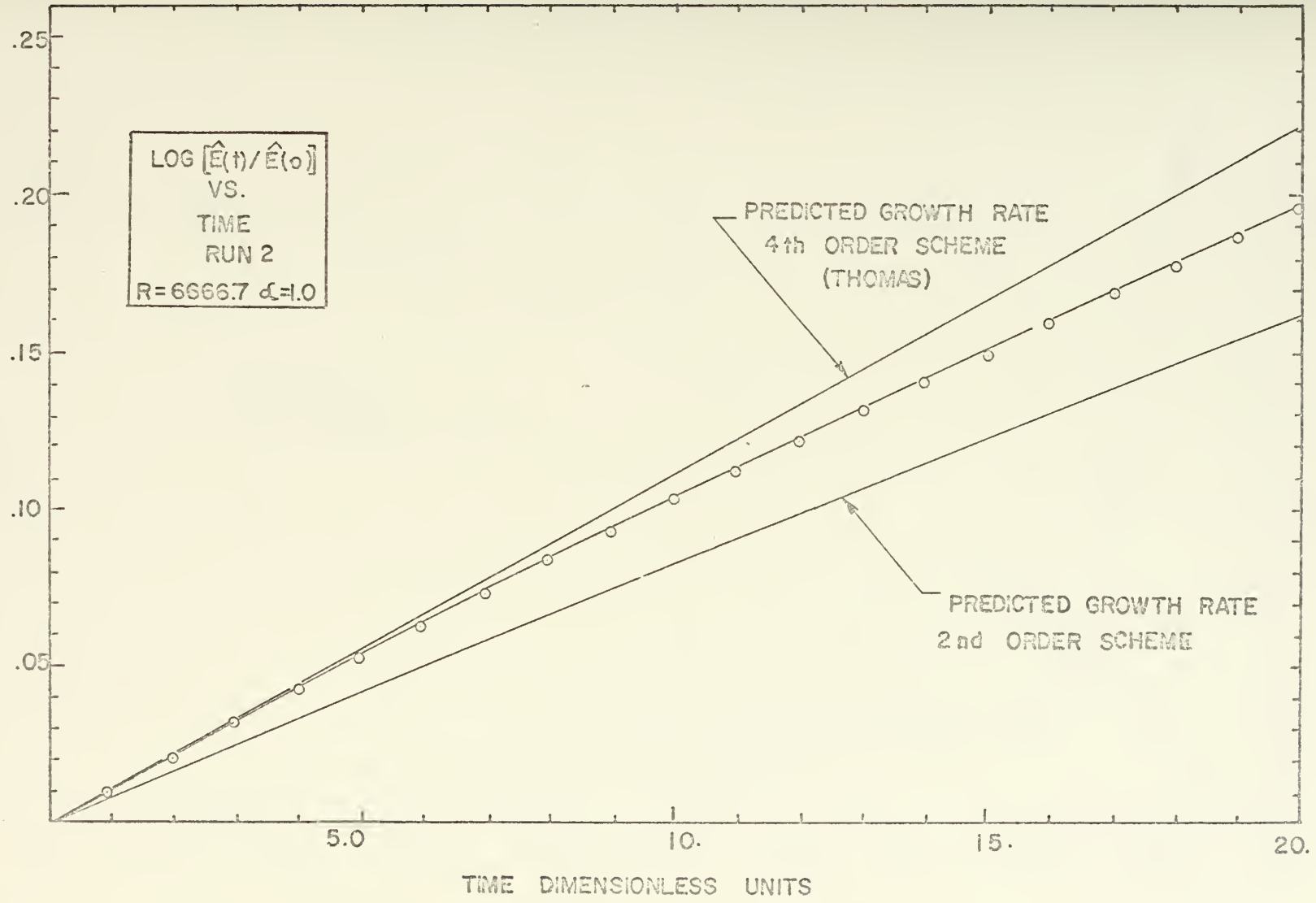


FIGURE 13 TEST FOR LINEAR SOLUTION.

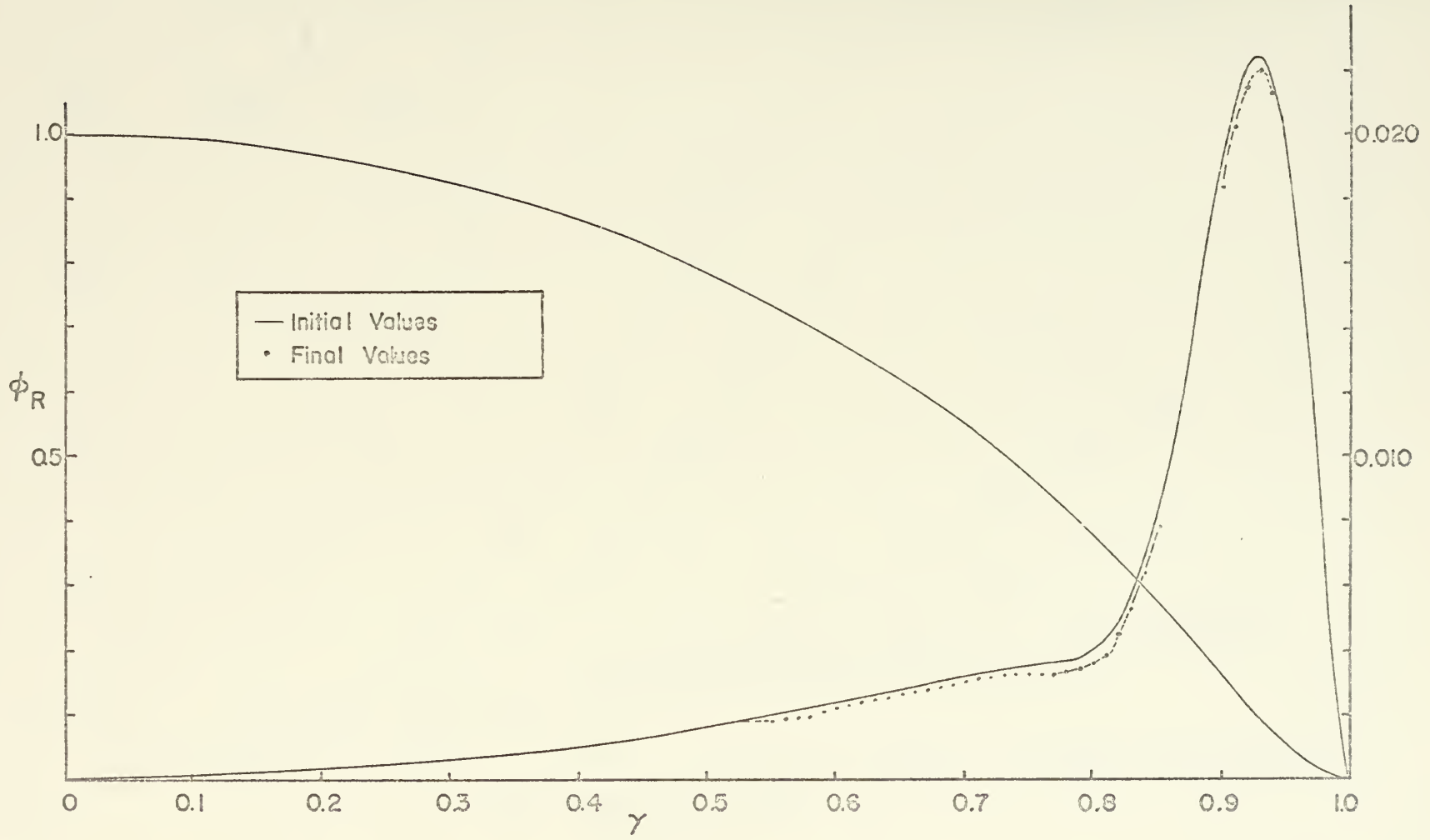


FIGURE 14 RUN NO. 2 DISTORTION OF EIGENVECTOR AFTER 1000 STEPS.

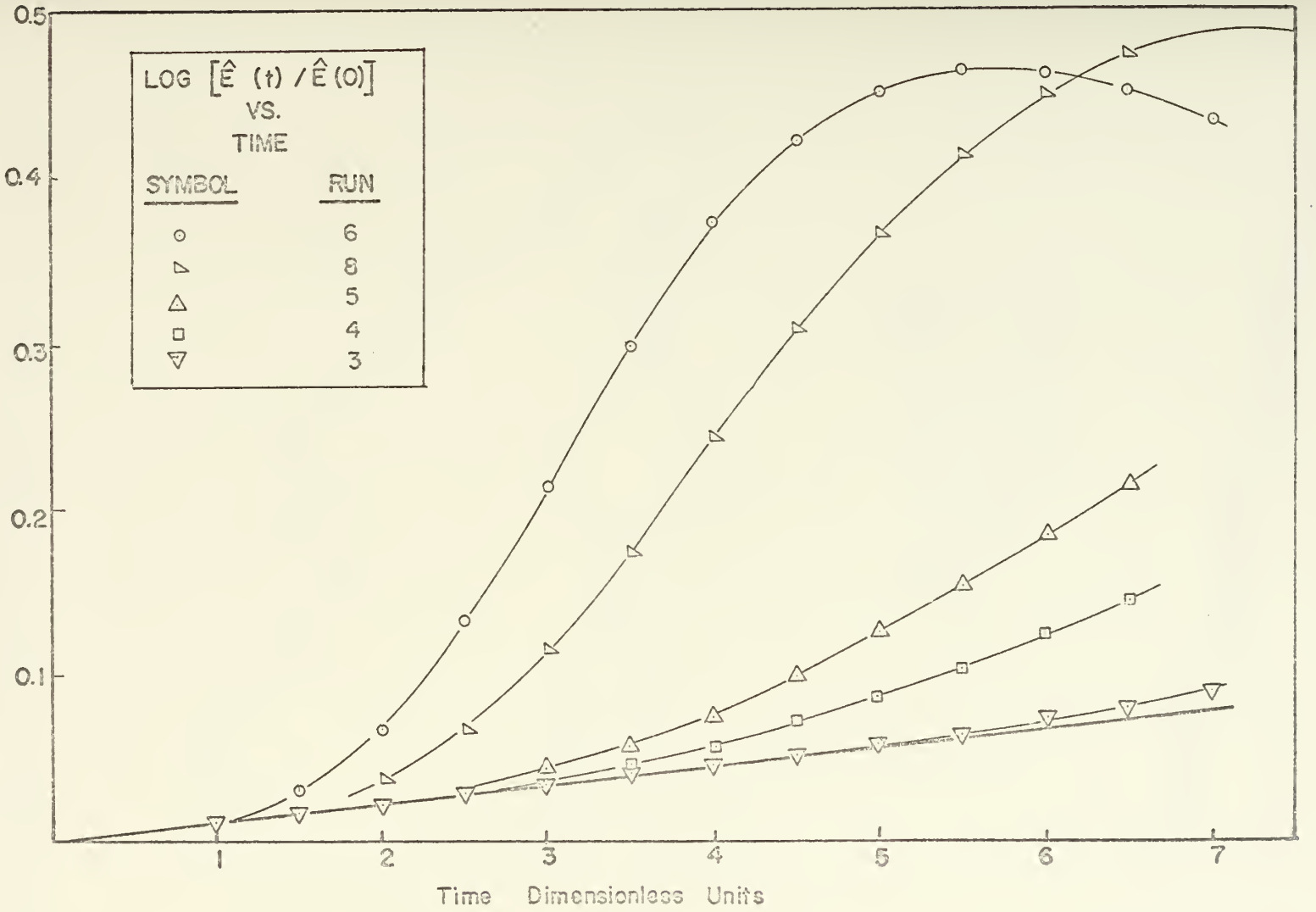


FIGURE 15
EFFECT OF AMPLITUDE ON INITIAL GROWTH RATE

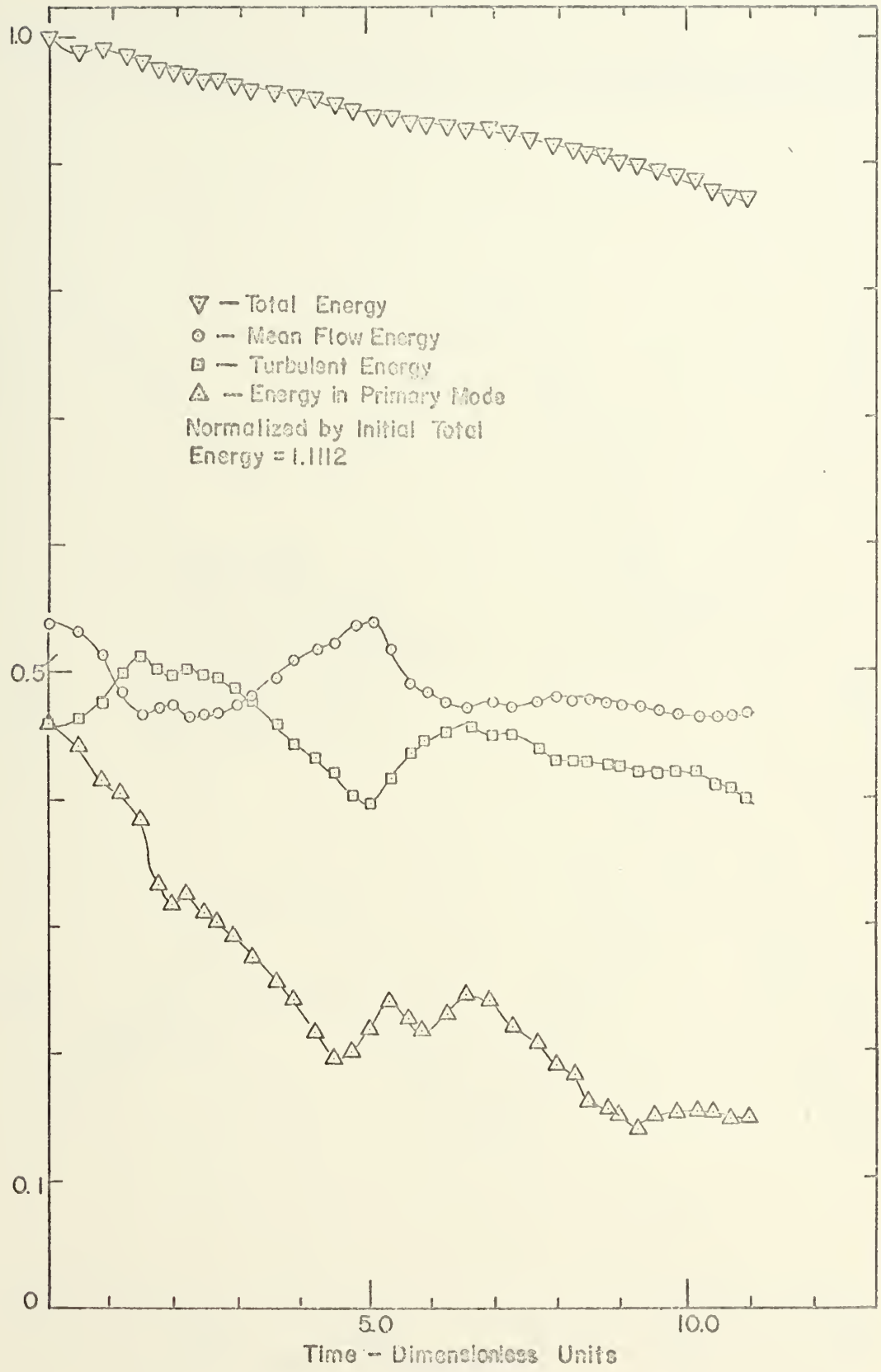


FIGURE 16 ENERGIES VS. TIME — RUN 7

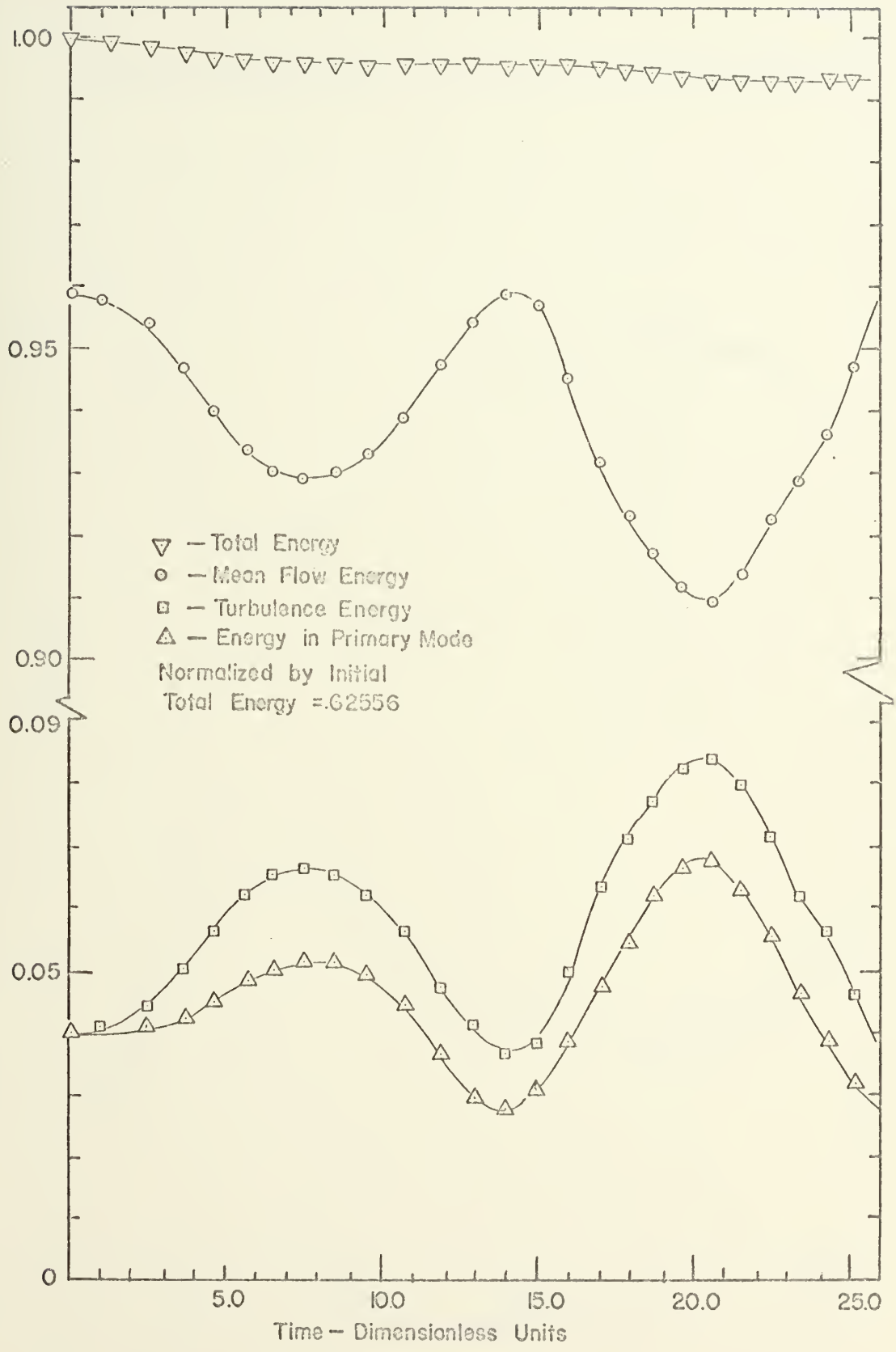


FIGURE 17A ENERGIES VS. TIME– RUN 8

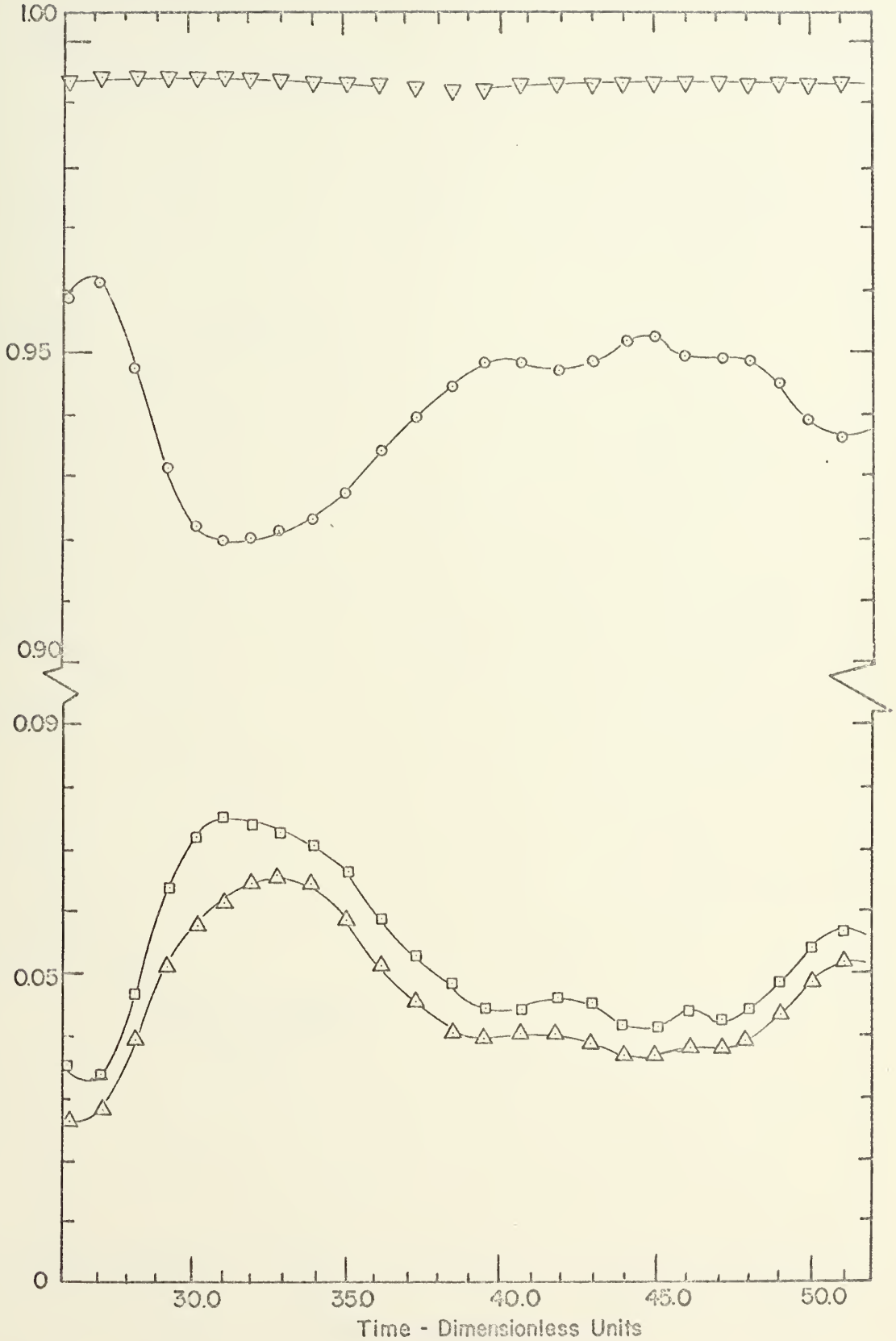


FIGURE 17B ENERGIES VS. TIME – RUN 8 Cont.

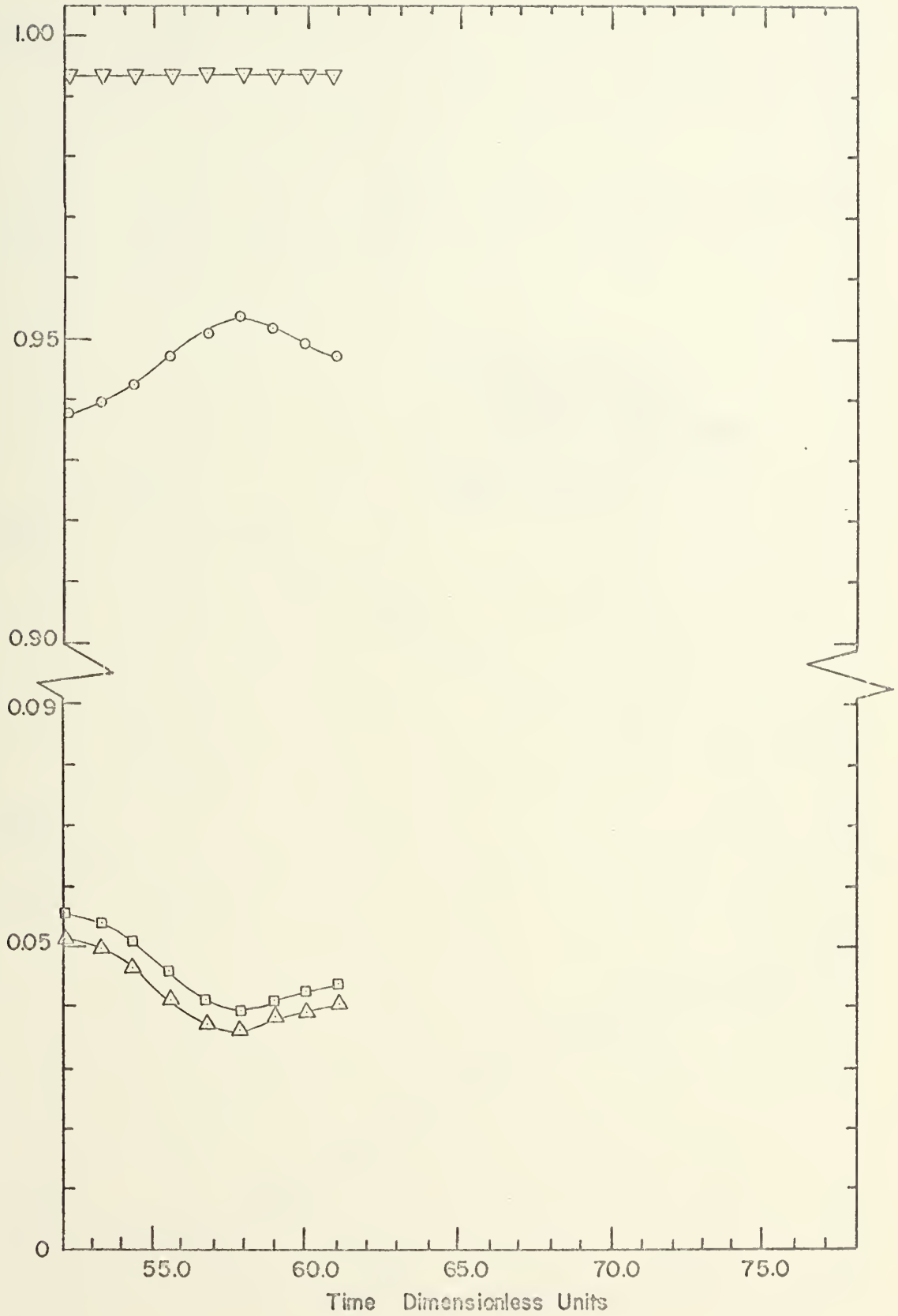


FIGURE 17C ENERGIES VS. TIME — RUN 8 Cont.

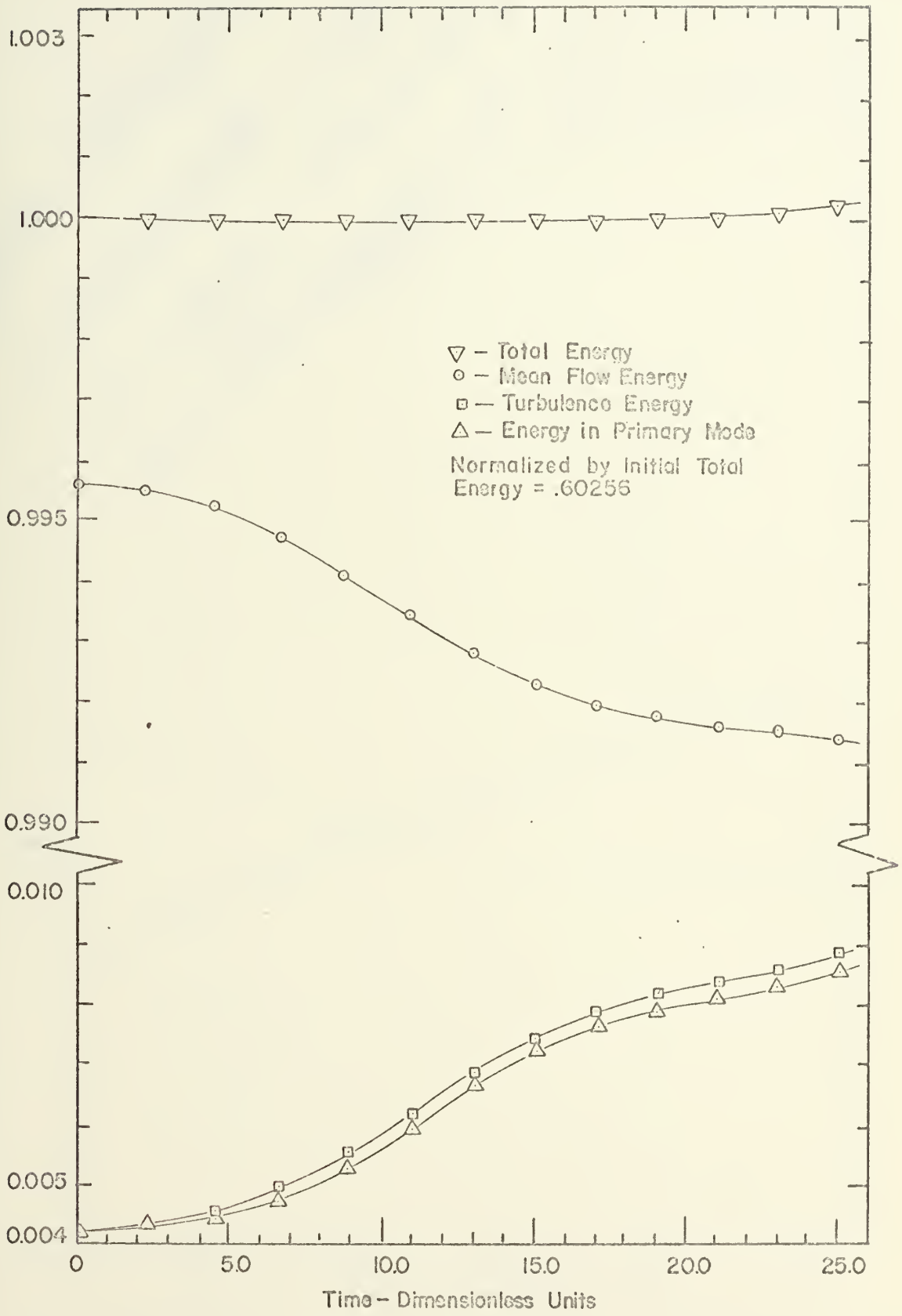


FIGURE 18A ENERGIES VS. TIME — RUN 9

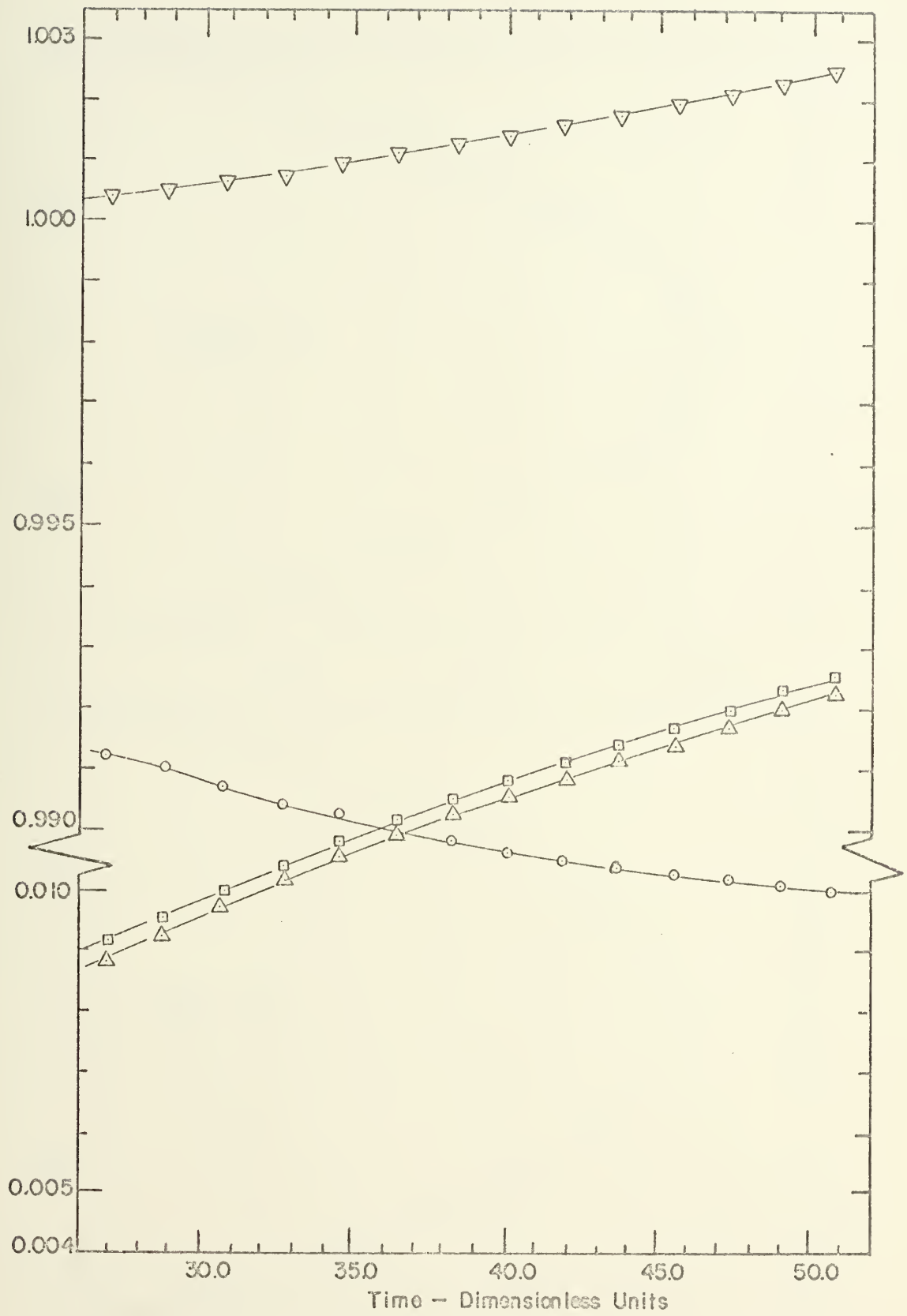


FIGURE 18B ENERGIES VS. TIME — RUN 9 Cont.

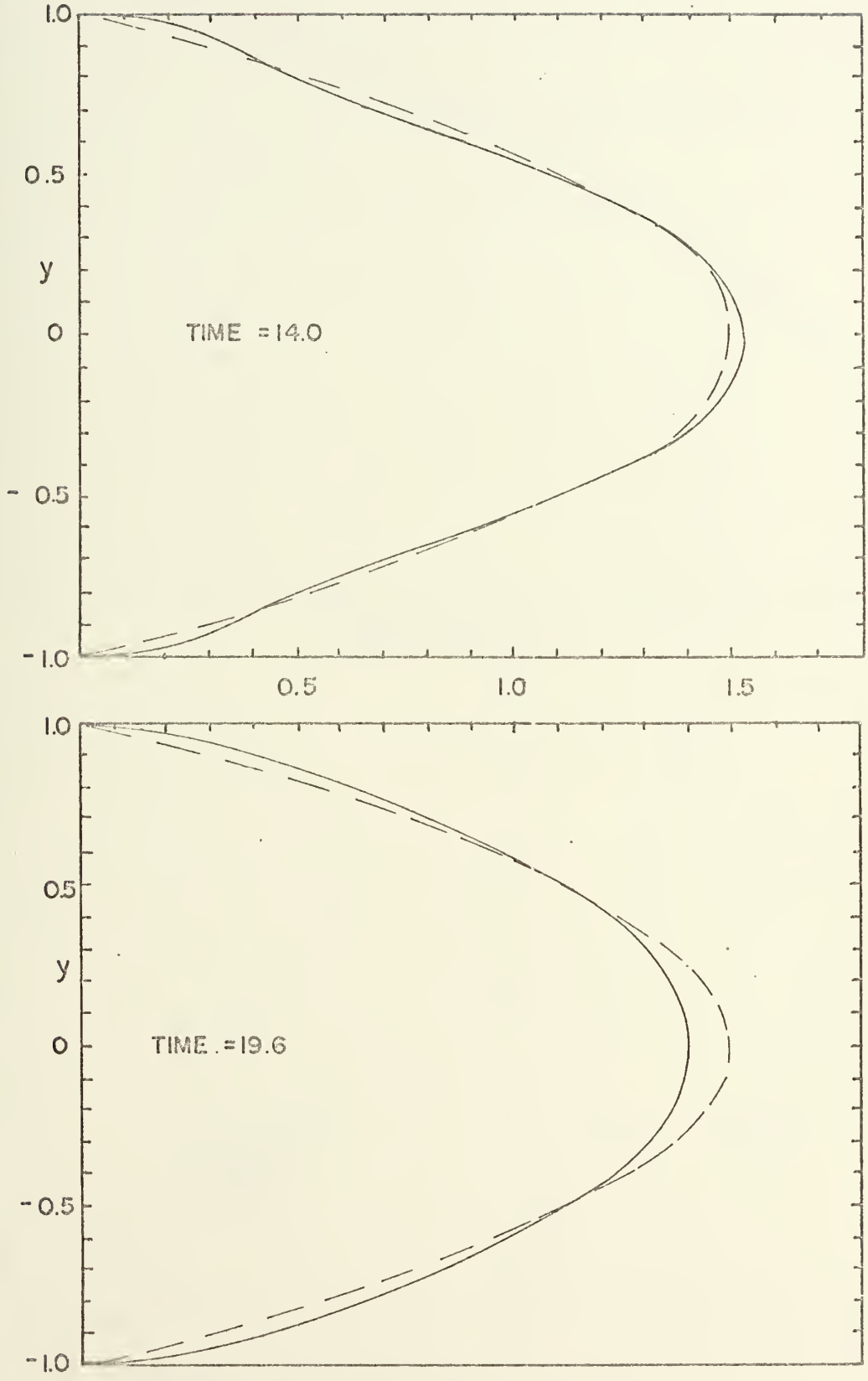


FIGURE 19 MEAN VELOCITY PROFILES, RUN 8

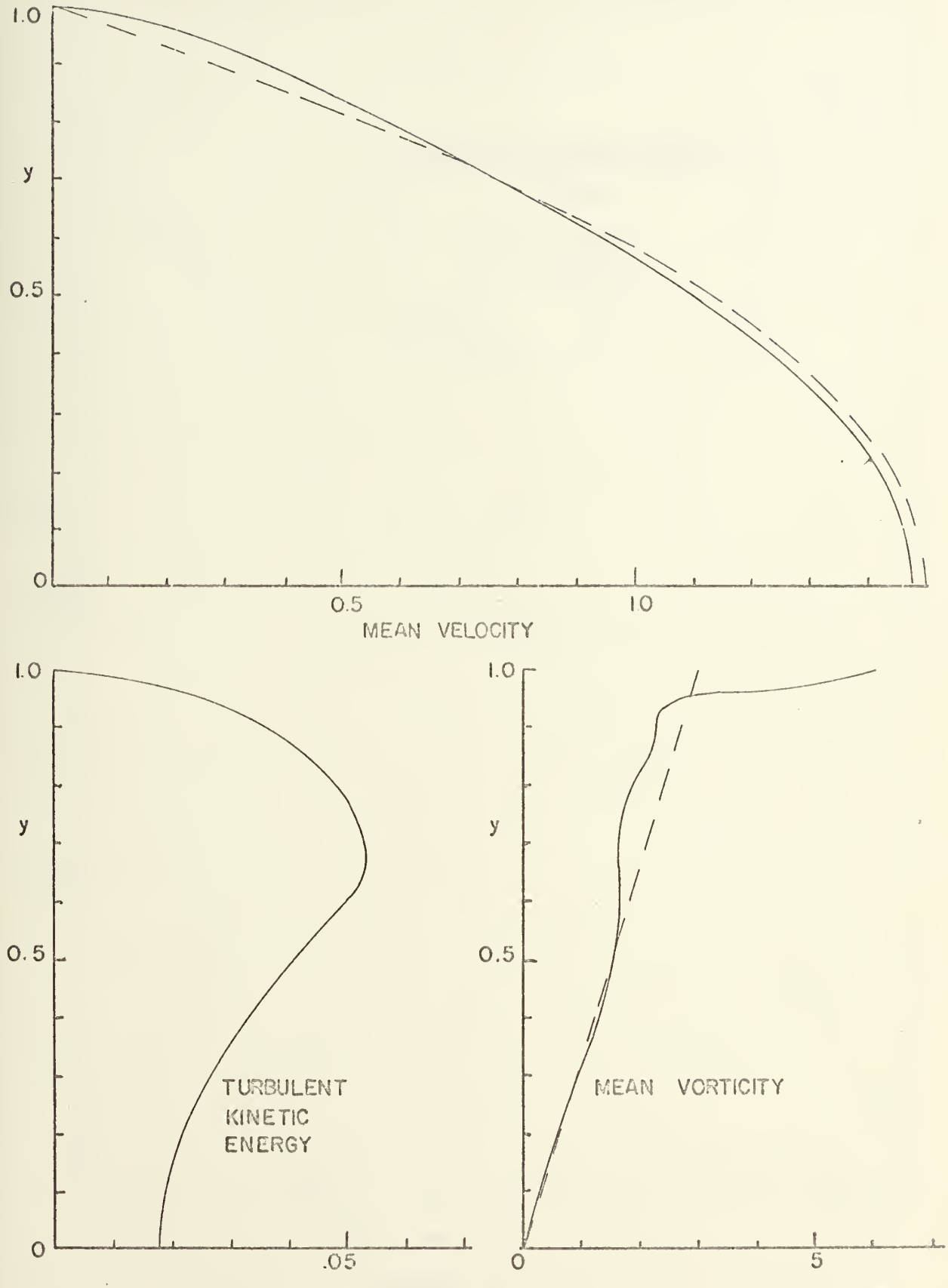


FIGURE 20. RUN 8. TIME = 53.2

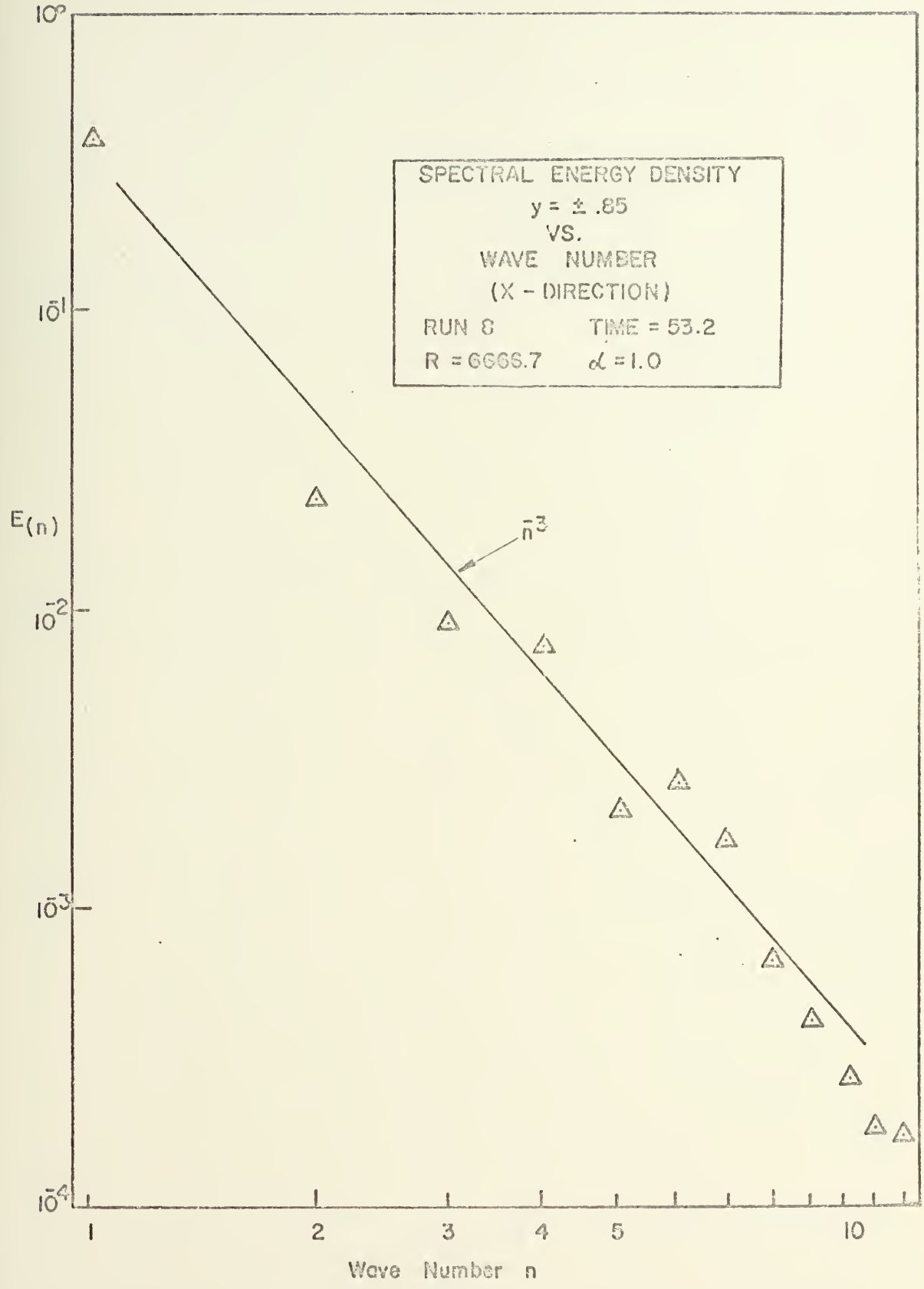

SPECTRAL ENERGY DENSITY
y = ± .85
VS.
WAVE NUMBER
(X - DIRECTION)
RUN 8 TIME = 53.2
R = 6666.7 α = 1.0
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
E(n)
n^{-3}
1
2
3
4
5
10
Wave Number n

FIGURE 21

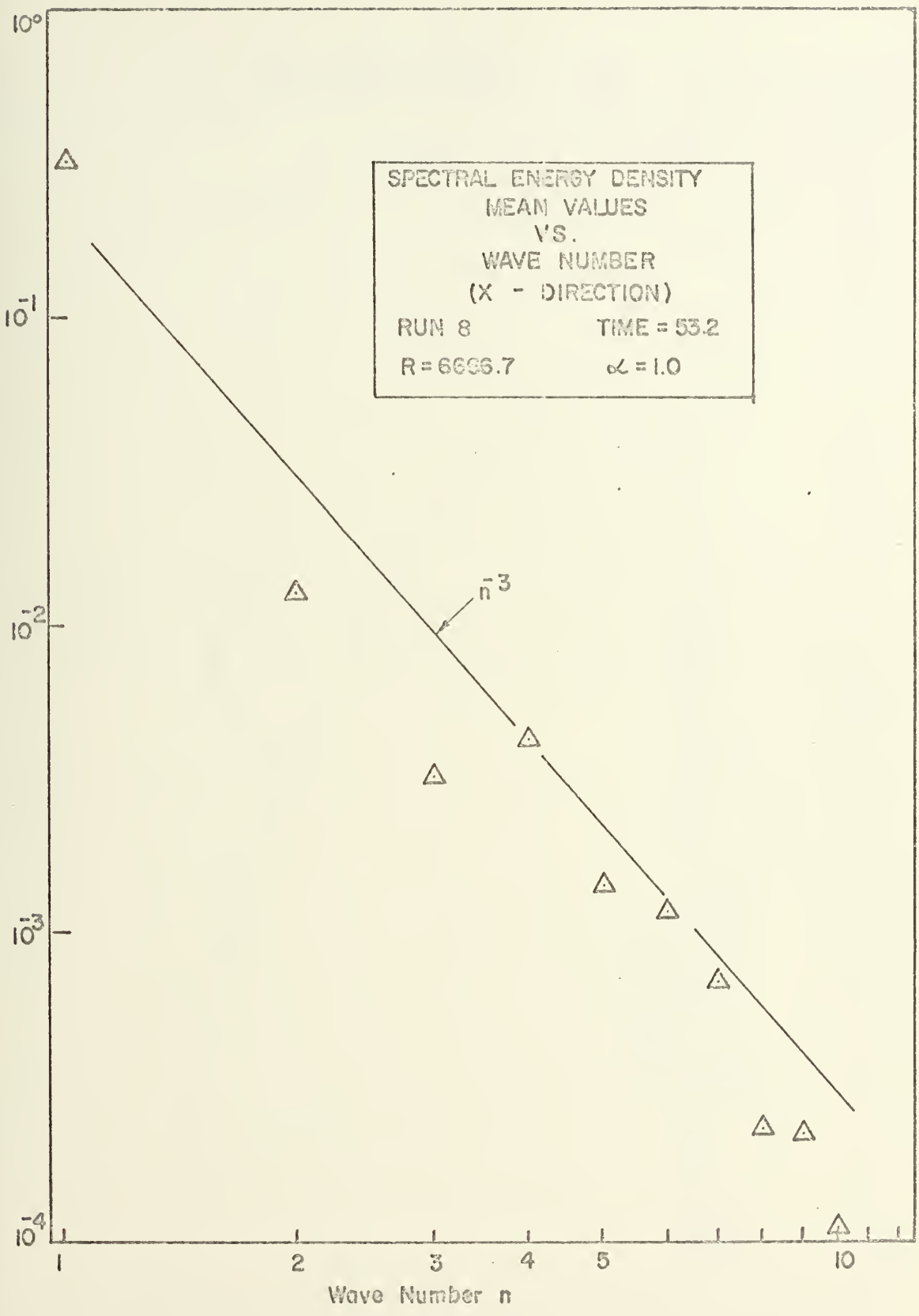

SPECTRAL ENERGY DENSITY
MEAN VALUES
V'S.
WAVE NUMBER
(X - DIRECTION)
RUN 8
TIME = 53.2
R = 6666.7
α = 1.0
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
n^{-3}
1
2
3
4
5
10
Wave Number n

FIGURE 22

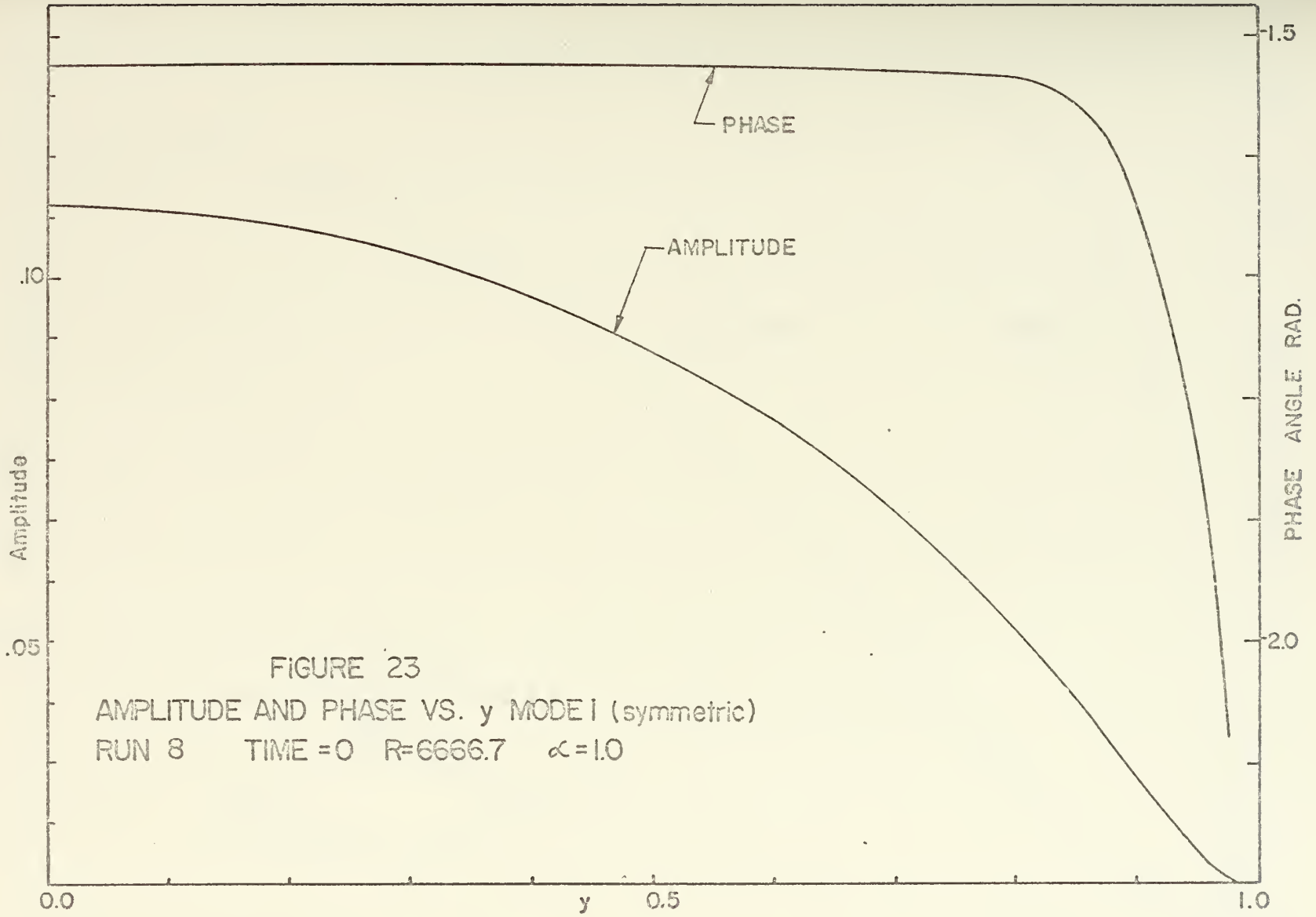


FIGURE 23

AMPLITUDE AND PHASE VS. y MODE I (symmetric)

RUN 8 TIME = 0 R=6666.7 α = 1.0

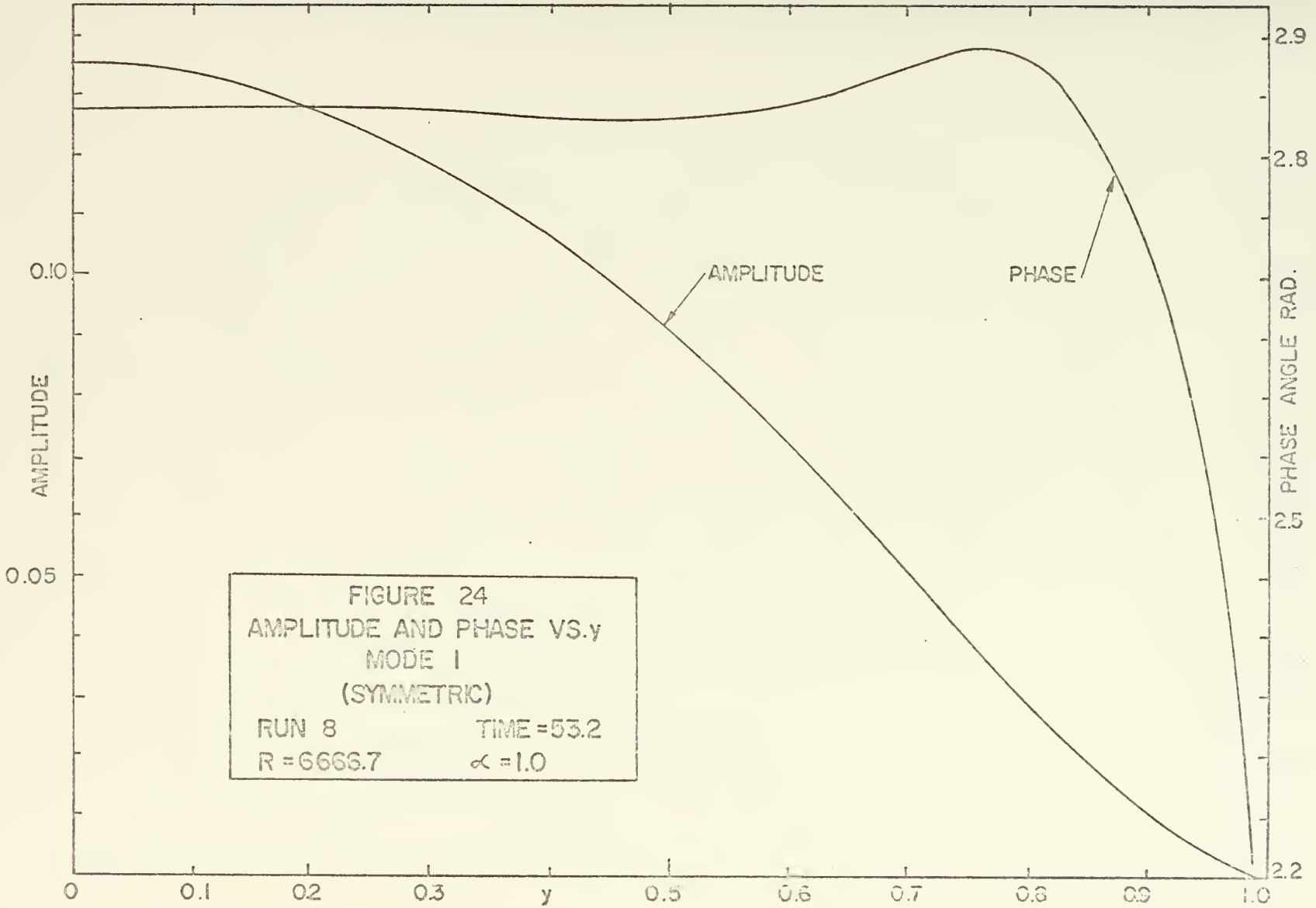
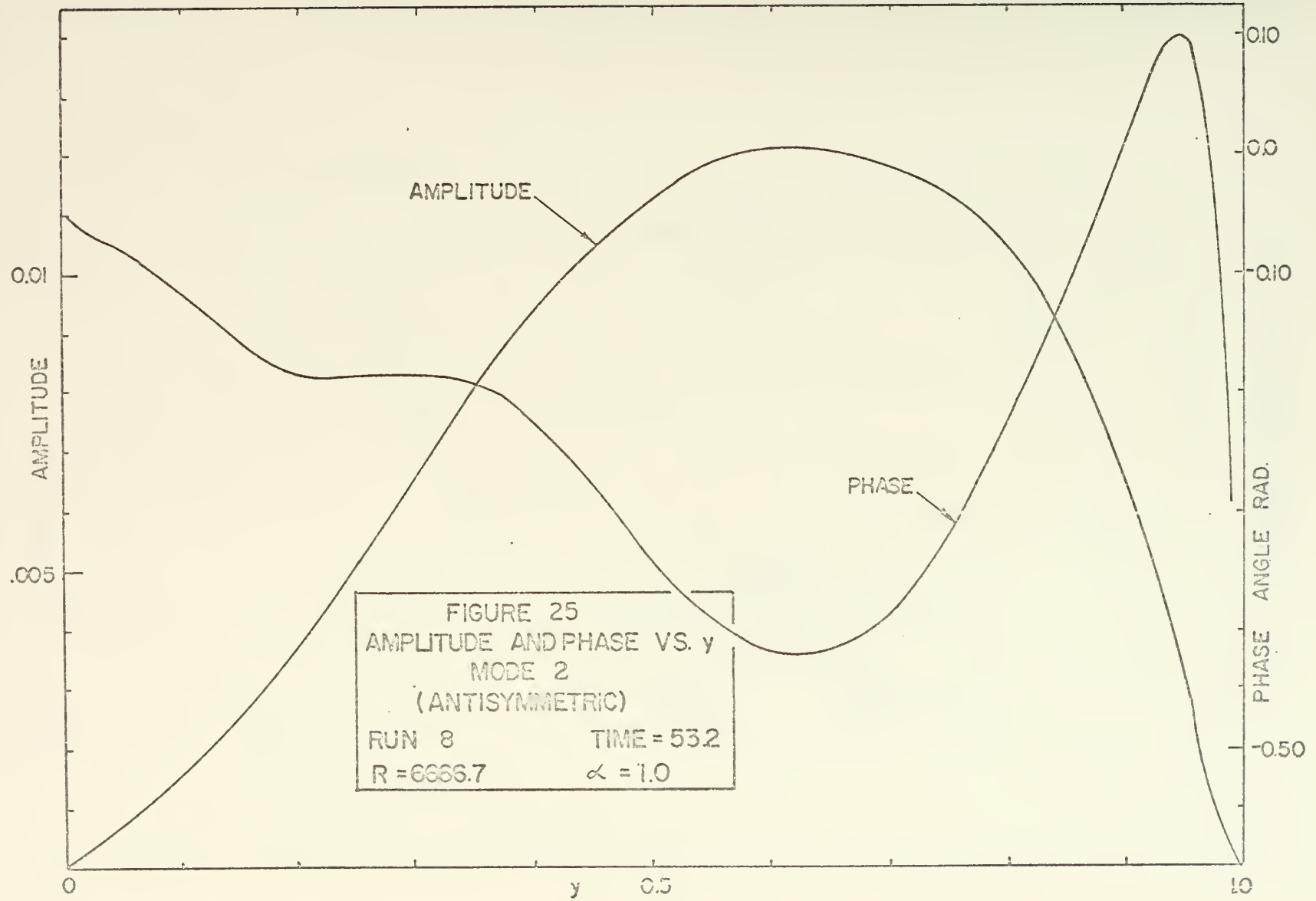


FIGURE 24
AMPLITUDE AND PHASE VS.y
MODE 1
(SYMMETRIC)
RUN 8 TIME = 53.2
R = 6666.7 α = 1.0


AMPLITUDE
PHASE
0.01
.005
AMPLITUDE
0.10
0.0
-0.10
-0.50
PHASE ANGLE RAD.
0
y
0.5
1.0
FIGURE 25
AMPLITUDE AND PHASE VS. y
MODE 2
(ANTISYMMETRIC)
RUN 8
TIME = 53.2
R = 6666.7
α = 1.0

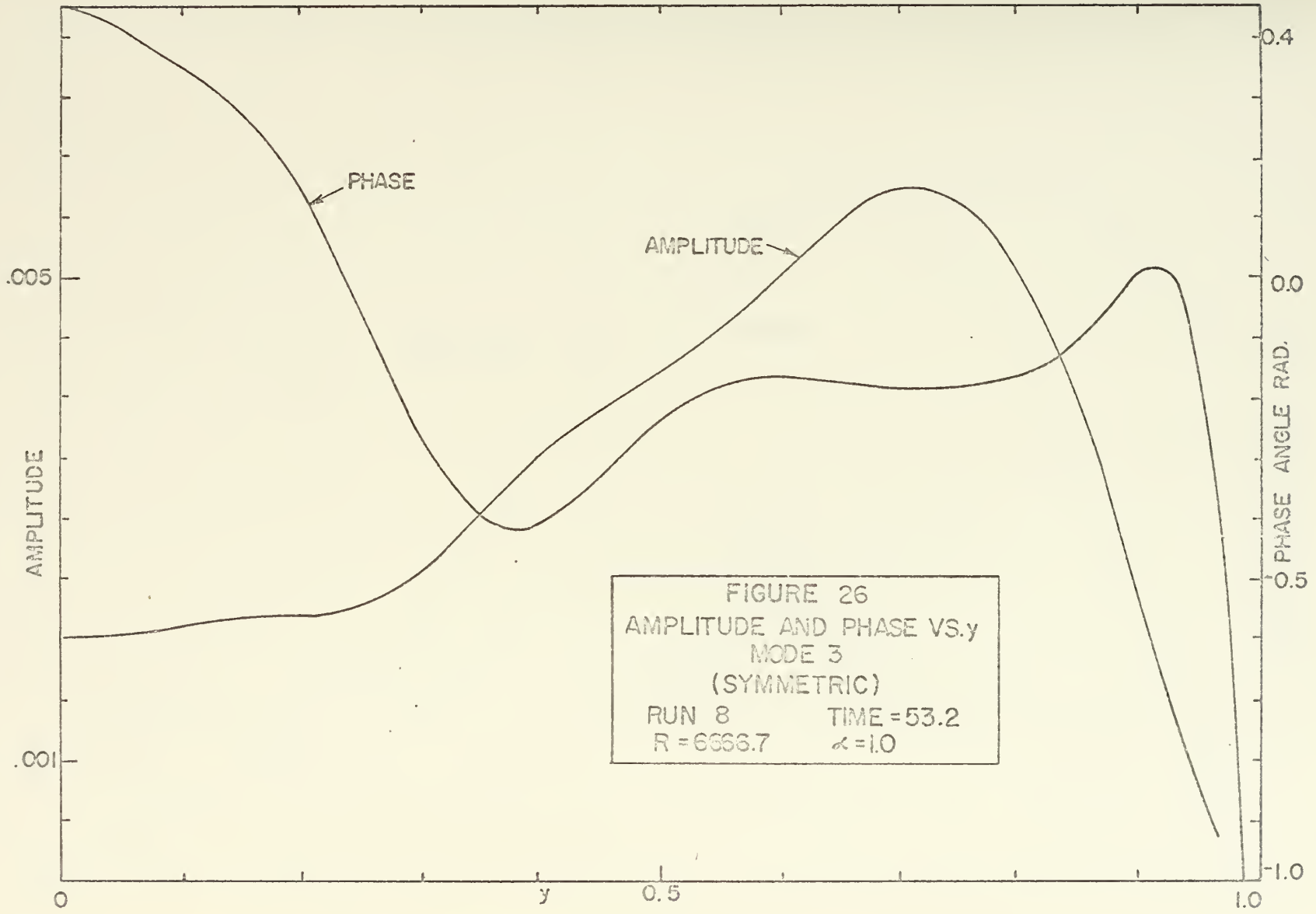
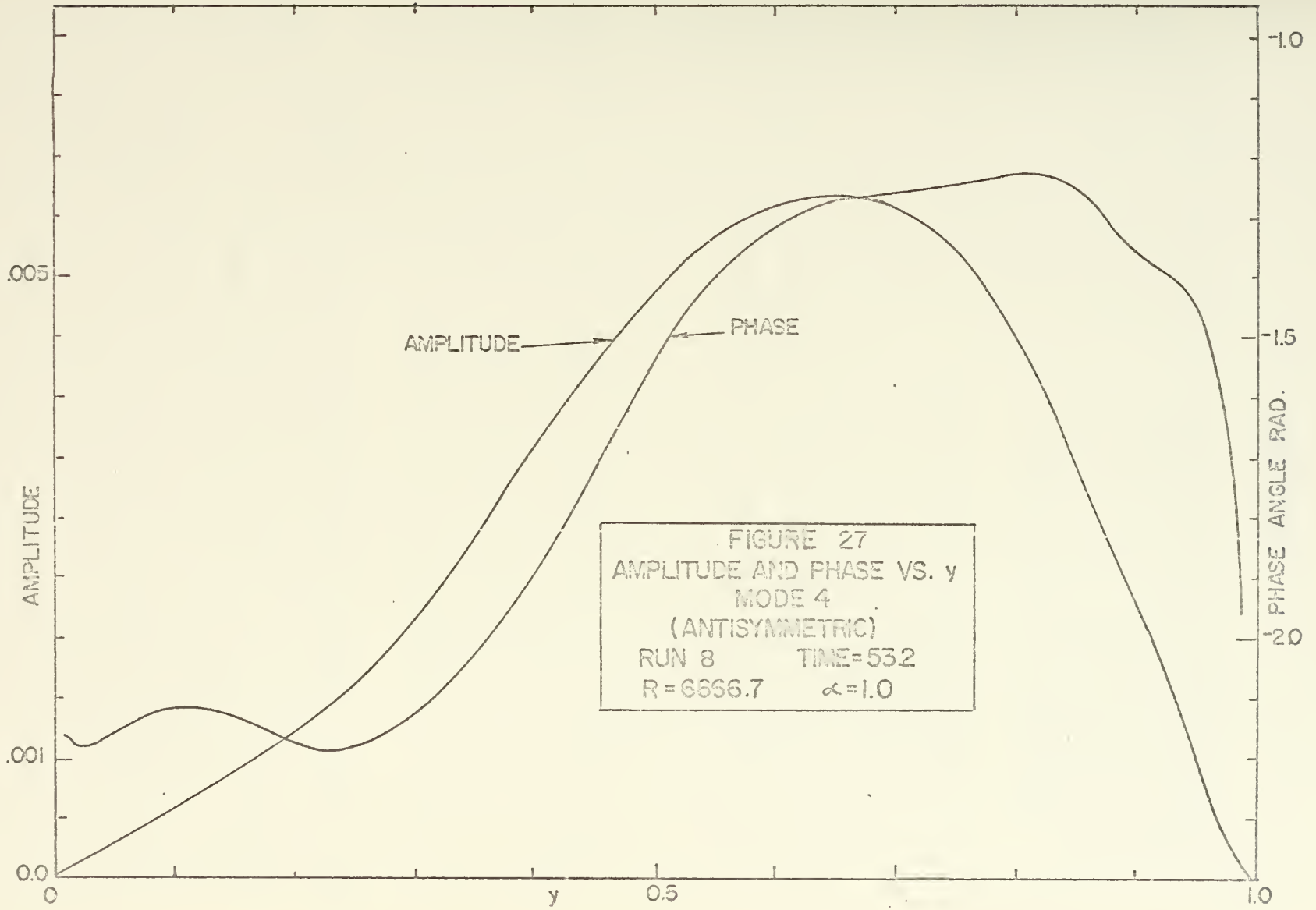


FIGURE 26
AMPLITUDE AND PHASE VS.y
MODE 3
(SYMMETRIC)
RUN 8 TIME = 53.2
R = 6666.7 α = 1.0


AMPLITUDE
PHASE
.005
.001
0.0
0
y
0.5
1.0
-1.0
-1.5
-2.0
PHASE ANGLE RAD.
AMPLITUDE
FIGURE 27
AMPLITUDE AND PHASE VS. y
MODE 4
(ANTISYMMETRIC)
RUN 8 TIME=53.2
R=6666.7 α=1.0

VI. CONCLUSIONS

The finite-difference representation of the vorticity transport equations, and the explicit treatment of the boundary values of the vorticity are consistent, stable and accurate. This has been demonstrated, in the linear range, by comparison with the established results of the linear theory.

The application of the discrete Fourier transform to the solution of the system of equations, associated with the representation of the Poisson equation, has produced significant improvements in both the speed and the accuracy of the calculations. Numerical tests indicate that the usual iterative schemes are inadequate if the number of mesh points is large.

Extensive calculations based on a consistent second-order representation of the linear eigenvalue problem provided a measure of the effect of mesh size on the accuracy of the representation of the large-scale unstable motions. The results indicate that the mesh sizes employed in previous investigations were often inadequate to represent linear instabilities.

Calculations in the non-linear range, at a moderate Reynolds number, indicate the existence of an equilibrium amplitude of the disturbance motion. In a long-term numerical integration, an approximate value of the equilibrium amplitude has been obtained. At the equilibrium state, the mean intensity of the two-dimensional turbulence is somewhat greater than measured values in real turbulent flows. This reflects the reduced dissipation due to the constraints on energy cascades in two-dimensional turbulence. The calculated spectral densities of kinetic energy are

consistent with the -3 power law predicted by Kraichnan for the two-dimensional turbulence.

The present program provides a means, of demonstrated accuracy, for the extensive investigation of finite-amplitude instability and two-dimensional turbulence.

VII. RECOMMENDATIONS FOR FUTURE WORK

The computer programs developed in the course of this investigation for both the solution of the linear eigenvalue problem and the numerical integration have yet to be fully exploited. A number of interesting applications remain. Among them are:

1. More extensive calculations for the solutions of the linear eigenvalue problem, in particular for the asymmetric modes. This will require only minor modifications of the program.
2. The development of an iterative procedure for the refinement of the solution for the eigenvector associated with the least stable mode. This will facilitate the investigation of finite-amplitude instabilities at sub-critical values of the Reynolds number.
3. A continuation of the integration for run number nine. The results to date show the disturbance increasing smoothly and monotonically at a decreasing rate. If, as seems likely, the energy levels off at the values associated with run eight, the indication of the equilibrium state will be more emphatic.
4. A more extensive investigation of the structure of the two-dimensional turbulent motion. This could be useful for the testing and development of heuristic models, in particular, the promising model developed by Gawain and Pritchett (1970).
5. Finally, some efforts should be made to further improve the speed of the calculations. A computer analysis of the operations in the program for the numerical integration indicates that substantial improvements in speed could be achieved by coding just Subroutine ADVANC in assembly language.

APPENDIX A

DERIVATION OF DISCRETE RELATIONS FOR THE SPECTRAL DENSITIES OF SQUARED VORTICITY AND KINETIC ENERGY

Associated with discrete Fourier transform pair, equations (4.46), is the convolution

$$\frac{1}{M}\sum_{n=1}^{M} T_n\, T_n^*\, e^{-i\,\theta_{n,s}} = \sum_{K=1}^{M} F_K\, F_{K-s+1}^* \qquad (A.1)$$

where $\theta_{n,s}$ is as given by equation (4.47). Equation (A.1) can be derived by substituting the appropriate transform or inverse relation, and reversing the order of the summation.

For s = 0 we have the spectral density relation

$$\frac{1}{M}\sum_{n=1}^{M} T_n\, T_n^* = \sum_{K=1}^{M} F_K\, F_K^* \qquad (A.2)$$

The mean value of squared vorticity is given by

$$a_J = \frac{1}{M}\sum_{K=1}^{M} \Gamma_{K,J}\, \Gamma_{K,J}^* \qquad (A.3)$$

Then from equations (A.3), (4.49) and (A.2),

$$a_j = \frac{1}{M^2}\sum_{n=1}^{M} G_{n,J}\, G_{n,J}^*$$

Since $\Gamma_{K,J}$ is real,

$$a_J = \frac{1}{M^2}\left\{ G_{1,J}^2 + 2\sum_{n=2}^{M/2} G_{n,J}\, G_{n,J}^* + G_{M/2+1,J}^2 \right\} \qquad (A.4)$$

Thus the quantities $G_{1,J}^2$, $2G_{2,J}\, G_{2,J}^*$, $2G_{3,J}$, $G_{3,J}^*$,..., etc. are the spectral components of mean squared vorticity. Notice that, in the present notation the first spectral term, n=1, refers to the contribution

of the disturbance to the mean flow. If the laminar flow vorticity is included then the first term becomes

$$(G_{1,J} + 3y_J)^2$$

Of interest also are the spectral components of the normalized sum of a_J over y,

$$\frac{1}{N-1}\left\{\frac{a_1}{2} + \sum_{J=2}^{N-1} a_J + \frac{a_N}{2}\right\}$$

which are obtained by performing the same operation on the various terms on the right of equation (A.3).

Expressions for the spectral components of kinetic energy, at a given value of J, or their normalized sums over y, can be derived in a similar way. The components of turbulent velocity can be expressed as

$$\hat{u}_{K,J} = \frac{1}{M}\sum_{n=1}^{M} U_{n,J}\, e^{-i\,\theta_{n,K}}$$

$$\hat{v}_{K,J} = \frac{1}{M}\sum_{n=1}^{M} V_{n,J}\, e^{-i\,\theta_{n,K}} \qquad \text{(A.5)}$$

where, from equations (2.25) and (4.49) $U_{n,J}$, and $V_{n,J}$ are given by

$$U_{n,J} = -\frac{F_{n,J+1} - F_{n,J-1}}{2\delta y}$$

$$V_{n,J} = -\frac{i \sin \theta_{n,2}}{\delta x} F_{n,J} \qquad \text{(A.6)}$$

Thus from equations (4.24) and (A.2) the mean value of turbulent kinetic energy is given by

$$\hat{E}_J = \frac{1}{2M} \sum_{n=2}^{M} U_{n,J}\, U^*_{n,J} + \frac{1}{2M} \sum_{n=2}^{M} V_{n,J}\, V^*_{n,J}$$

Since $u_{K,J}$ and $v_{K,J}$ are real

$$\hat{E}_J = \frac{1}{2M}\left\{ 2 \sum_{n=2}^{M/2} \left(U_{n,J}\, U^*_{n,J} + V_{n,J}\, V^*_{n,J} \right) + U^2_{M/2+1,J} + V^2_{M/2+1,J} \right\} \tag{A.7}$$

Finally, the spectral components for

$$\hat{E} = \frac{1}{N-1} \sum_{J=2}^{N-1} \hat{E}_J$$

are obtained by performing the same operation on the terms on the right in equation (A.7).

APPENDIX B

LISTING OF THE FORTRAN CODE FOR THE NUMERICAL INTEGRATION

```
      COMMON / FIELDS / PHI,GAMA,FG,GAMOLD                              CCCCCCE0
      COMMON / VECTOR / T,AS,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,    CCCCCCF0
     1                  U,PHILAM,EE,TA,GA,X,Y                          CCCCCD00
     1 COMMON / ARRAYS / IPO,IINV,IIAP,NM(3)                           CCCCCD10
      COMMON / PARAMS / N,M,NP1,NP2,NM1,ND2,NM3,NMA,NH,MP1,MP2,MP3,MM1, CCCCCD20
     1                  N2M,NH1,MHP,NPRCB,ICOUNT,ITEL,INT,KSTEP,ISTART, CCCCCD30
     1                  ISCIP,ITAPE,NITER,NITOT,NITMAX,IFIC,MCOMP,     CCCCCD40
     1                  KINT                                           CCCCCD50
     2 COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, CCCCCD60
     2                  RSQRP,RELAND,RELAYLAND,TIME,DT,DTT,PI,H,       CCCCCD70
     2                  REKMEAN,RELATER,RMALPHA,BETA,XM,XN             CCCCCD80
     2                  EKMEAN,RELKTUR,RUVSQR,ATA,EC,F1,F2,F3,F4,F5,F6 CCCCCD90
     3 COMMON / SYMBOL / DOT,BLANK,XXX                                 CCCCCE00
      DOUBLE PRECISION   DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR CCCCCE10
     1                  RSQRP,RELAND,RELAYLAND,TIME,DT,DTT,PI,H,       CCCCCE20
     1                  EKMEAN,RELKTUR,RUVSQR,ATA,EC,F1,F2,F3,F4,F5,F6 CCCCCE30
     3 DIMENSION PHI(64,4),XM(20),GAMA(20),TURBKE(20),UMEAN(20),       CCCCCE40
      DOUBLE PRECISION   PHMEAN(20),PHILAM(20),GAMEAN(20),X(64),Y(20),F(32), CCCCCE50
     1                  PHI,GAMA,XM,TURBKE,UMEAN,PHMEAN,PHILAM,GAMEAN,X,Y,F CCCCCE60
      DIMENSION          ETA(20),GA(20),INV(32),IAP(64),IAP(64)        CCCCCE70
      REAL*4 TITL(24)/'INITIAL VALUES OF PHI'/                          CCCCCE80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCE90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCF90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCG90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCH90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCI90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCJ90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCK90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCL90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCM90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCN90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCO90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCP90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCQ90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCR90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCS90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCT90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCU90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCV90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCW90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCX90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCY90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ00
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ10
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ20
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ30
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ40
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ50
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ60
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ70
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ80
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCCZ90
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC100
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC110
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC120
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC130
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC140
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC150
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC160
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC170
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC180
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC190
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC200
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC210
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC220
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC230
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC240
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC250
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC260
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC270
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC280
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC290
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC300
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC310
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC320
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC330
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC340
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC350
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC360
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC370
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC380
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC390
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC400
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC410
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC420
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC430
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC440
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC450
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC460
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC470
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC480
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC490
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC500
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC510
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC520
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC530
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC540
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC550
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC560
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC570
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC580
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC590
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC600
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC610
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC620
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC630
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC640
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC650
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC660
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC670
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC680
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC690
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC700
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC710
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC720
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC730
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC740
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC750
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC760
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC770
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC780
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC790
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  *  CCCCC800
     *  *  *  *  *  *  *  *  *  *  *  *  *  *  
```

```
C     DIMENSIONS OF ARRAYS                                                    COOOOC530
C      PHI,GAMA,FG,GAMOLD --- MXN                                             COOOOC540
C      UMEAN,TEMP,TURBKE,U,PHILAM,T,PHMEAN,GAMEAN,Y --- N                     COOOOC550
C      XALPHA,X,IPO,IMO,IAO,IAP --- M                                         COOOOC560
C      S,INV --- M/2                                                          COOOOC570
C      A --- 2M                                                               COOOOC580
C                                                                             COOOOC590
C     THE FOLLOWING ARRAYS CONTAIN FIXED QUANTITIES AND SHOULD NEVER          COOOOC600
C      BE USED FOR TEMPORARY STORAGE.                                         COOOOC610
C                                                                             COOOOC620
C      U,PHILAM,S,XALPHA,X,Y,IPO,IMO,INV,IAP,IAO                              COOOOC630
C                                                                             COOOOC640
C      OTHER ARRAYS MAY BE USED FOR TEMPORARY STORAGE AT CERTAIN PARTS        COOOOC650
C      OF THE PROGRAM.ARRAYS T AND A ARE MOST SUITABLE FOR THIS               COOOOC660
C      PURPOSE.                                                               COOOOC670
C                                                                             COOOOC680
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *     COOOOC690
C                                                                             COOOOC700
C                                                                             COOOOC710
C                                                                             COOOOC720
C                                                                             COOOOC730
C     SUBROUTINE          FUNCTION                                            COOOOC740
C                                                                             COOOOC750
C      SETUP              ESTABLISH VALUES OF VARIOUS QUANTITIES WHICH        COOOOC760
C                         REMAIN CONSTANT THROUGHOUT THE INTEGRATION          COOOOC770
C      START              ESTABLISH INITIAL VALUES OF STREAM FUNCTION         COOOOC780
C                         AND VORTICITY                                       COOOOC790
C      TITLE              WRITES OUTPUT TITLE                                 COOOOC800
C      STEP               TIME STEP BY MODIFIED EULER IMPLICIT TIME           COOOOC810
C                         DIFFERENCING                                        COOOOC820
C      ADVANC             TIME STEP BY DUFORT FRANKEL CENTRAL DIFFERENCING    COOOOC830
C      PRESUR             SOLVES DELSQUARE(PHI)=GAMA FOR PHI BY FINITE        COOOOC840
C                         FOURIER TRANSFORM                                   COOOOC850
C      TRISOL             SOLUTION OF A SPECIAL TRIDIAGONAL SYSTEM OF         COOOOC860
C                         LINEAR EQUATIONS BY GAUSS ELIMINATION.              COOOOC870
C                         CALLED BY SUBROUTINE PRESUR                         COOOOC880
C      TIMER              COMPUTES A NEW VALUE OF DT                          COOOOC890
C      MEANS              COMPUTES MEAN VALUES OF DISTURBANCE STREAM          COOOOC900
C                         FUNCTION AND VORTICITY ---PHMEAN AND GAMEAN         COOOOC910
C      ENERGY             COMPUTES THE MEAN KINETIC ENERGY OF THE             COOOOC920
C                         TURBULENCE AS A FUNCTION OF Y, THE INTEGRAL         COOOOC930
C                         OVER Y OF THAT QUANTITY AND THE MEAN KINETIC        COOOOC940
C                         ENERGY OF THE MEAN FLOW                             COOOOC950
C      EDDYS              COMPUTES TURBULENCE STREAM FUNCTION AND             COOOOC960
C                         VORTICITY.                                          COOOOC970
C                         CALL EDDYS(+1) --- PUTS PHI-PHMEAN IN FG            COOOOC980
C                         CALL EDDYS(-1) --- PUTS GAMA-GAMEAN IN FG           COOOOC990
C      STRESS             COMPUTES TURBULENT REYNOLDS STRESS                  COOOIOOO
```

```
C       PRINT              PRINTS OUTPUT LINE                                    CCCC101C
C       PLOT               PRINTS ONLINE GRAPHS OF MEAN VELOCITY,VORTICITY       CCCC1C20
C                          TURBULENT KINETIC ENERGY AND REYNOLDS STRESS          CCCC1C30
C       SPCTRM             COMPUTES SPECTRAL DENSITY OF TURBULENT                CCCC1040
C                          KINETIC ENERGY OR SQUARED VORTICITY AS                CCCC1050
C                          FUNCTIONS OF Y ,AND THE MEAN OVER Y.                  CCCC1060
C                          CALLED BY PRESUP                                      CCCC1070
C                          CALL SPCTRM(1) --- KINETIC ENERGY                     CCCC1080
C                          CALL SPCTRM(-1) --- SQUARED VORTICITY                 CCCC1090
C       MODES              COMPUTES AND PRINTS MODAL VECTORS                     CCCC1100
C                          CALL MODES(MOD1,MOD2,1) PRINTS VECTORS FOR            CCCC111C
C                          FOURIER MODES MOD1 AND MOD2 NORMALIZED                CCCC1120
C                          SO THAT AT Y=0 REAL PART = 1 .AND IMAGINARY           CCCC1130
C                          PART=0. CALL MODES(MOD1,MOD2,2) PRINTS                CCCC114C
C                          MODULUS AND PHASE                                     CCCC1150
C       PLOTSP             PRINTS ONLINE GRAPHS --- GENERAL                      CCCC1160
C                          CALL PLOTSP(A,B,N) PRINTS GRAPHS OF                   CCCC1170
C                          POSITIVE N-DIMENSIONAL VECTORS A AND B                CCCC1180
C       CPLOT              PRINTS CONTOUR PLOT                                   CCCC119C
C       OUTPUT             PRINTS TWO DIMENSIONAL ARRAYS.                        CCCC1200
C                          CALL OUTPUT(1) --- PRINTS PHI                         CCCC1210
C                          CALL OUTPUT(2) --- PRINTS GAMA                        CCCC1220
C                          CALL OUTPUT(3) --- PRINTS EG                          CCCC1230
C       LOAD               READS DATA FROM TAPE OR DISC                          CCCC1240
C       RELOAD             WRITES DATA ON TAPE OR DISC                           CCCC1250
C       DHARM              COMPUTES FINITE FOURIER TRANSFORM.                    CCCC1260
C                          DHARM IS INCLUDED IN THE IBM SCIENTIFIC LIBRARY       CCCC1270
C                          OF PRE COMPILED PROGRAMS AVAILABLE AT NPS.            CCCC128C
C                                                                                CCCC1290
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *    CCCC1300
C                                                                                CCCC1310
C     LOGICAL INDICES ----------------                                           CCCC1320
C      ICOUNT              NUMBER OF COMPLETED TIME STEPS                        CCCC1330
C      INT                 CONTROLS DETAILED OUTPUT---WHEN ICOUNT=INT            CCCC134C
C                          MAIN CALLS PRINT, PRESUP CALLS SPCTRM ETC.            CCCC1350
C      ISCIP               INCREMENT FOR INT                                     CCCC1360
C      MCOMP               CONTROLS SINGLE LINE OUTPUT---WHEN ICOUNT=MCOMP       CCCC1370
C                          MAIN CALLS PRINT                                      CCCC1380
C      MSCIP               INCREMENT FOR MCOMP                                   CCCC1390
C      KSTEP               CONTROLS CALL OF SUBROUTINE STEP. IF ICOUNT=          CCCC1400
C                          KSTEP TIME STEP IS PERFORMED BY STEP                  CCCC141C
C      KINT                INCREMENT FOR KSTEP.IF KINT=1 THEN INTEGRATION        CCCC142C
C                          IS BY MOD EULER TIME DIFFERENCING.                    CCCC143C
C                                                                                CCCC144C
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *    CCCC145C
C                                                                                CCCC1460
C                                                                                CCCC1470
      READ(5,11)NPROB,NCYCLS                                                     CCCC1480
```

```
      ITAPE=10                                                          COCO1490
      CALL ERRSET(255,1000,-1,1)                                        COCO1500
      IF(NPROB .LT. 0)GO TO 2                                           COCO1510
    1 READ(5,12) N,M,ISCIP,MSCIP                                        COCO1520
      READ(5,13) REYNLD,FRACEL,XLAMDA                                   COCO1530
      READ(5,19) DOT,BLANK,XXX                                          COCO1540
      INT=0                                                             COCO1550
      ISTART=0                                                          COCO1560
      KSTEP=0                                                           COCO1570
      MCOMP=0                                                           COCO1580
      TIME=0.                                                           COCO1590
      NITOT=0                                                           COCO1600
      CALL SETUP                                                        COCO1610
      CALL START                                                        COCO1620
      WRITE(6,14)NPROB,NCYCLS,N,M,NITMAX,ISCIP,MSCIP                    COCO1630
      GO TO 3                                                           COCO1640
    2 CALL LOAD                                                         COCO1650
      MCOMP=ISTART                                                      COCO1660
    3 CALL TITLE                                                        COCO1670
      WRITE(6,15)                                                       COCO1680
      WRITE(6,16)                                                       COCO1690
      IFIG=1                                                            COCO1700
      CALL PRINT                                                        COCO1710
      CALL PLOT                                                         COCO1720
C                                                                       COCO1730
      DO 10 ICOUNT=ISTART,NCYCLS                                        COCO1740
      IF(ICOUNT .NE. MCOMP) GO TO 5                                     COCO1750
    4 MCOMP=MCOMP+MSCIP                                                 COCO1760
      CALL PRINT                                                        COCO1770
    5 IF(ICOUNT .NE. INT) GO TO 7                                       COCO1780
    6 INT=INT+ISCIP                                                     COCO1790
      CALL PLOT                                                         COCO1800
    7 CALL ADVANC                                                       COCO1810
   10 CONTINUE                                                          COCO1820
      ISTART=NCYCLS+1                                                   COCO1830
      CALL PRINT                                                        COCO1840
      WRITE(6,17)                                                       COCO1850
      WRITE(6,18)                                                       COCO1860
   11 FORMAT(2I10)                                                      COCO1870
   12 FORMAT(5I10)                                                      COCO1880
   13 FORMAT(5G12.0)                                                    COCO1890
   14 FORMAT(7I10)                                                      COCO1900
   15 FORMAT(10X,'INITIAL VALUES OF STREAM FUNCTION'//)                 COCO1910
   16 FORMAT(10X,'INITIAL VALUES OF VORTICITY'//)                       COCO1920
   17 FORMAT(10X,'STREAM FUNCTION'//)                                   COCO1930
   18 FORMAT(10X,'VORTICITY'//)                                         COCO1940
   19 FORMAT(3A1)                                                       COCO1950
      CALL RELOAD                                                       COCO1960
```

```
      STOP                                                              00001970
      END                                                               00001980
```

```
      SUBROUTINE SETUP                                                  00001990
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              00002000
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00002010
     1                  U,PHILAM,EE,ETA,GA,X,Y                         00002020
      COMMON / ARRAYS / IPO,IMO,INV,IAO,IAP,MM(3)                       00002030
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 00002040
     1                  M2M,MH,MHP,NPOOR,ICOUNT,ITEL,INT,KSTEP,ISTART, 00002050
     2                  ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IEIG,MCOMP,00002060
     3                  KINT                                            00002070
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR3,DY,DY2,DYSQR,R,RSQR,  00002080
     1                  RSQRR,REYNLD,REY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H,00002090
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00002100
     3                 ,XNM1,UMNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6 00002110
      COMMON / SYMBOL / DOT,BLANK,XXX                                   00002120
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR3,DY,DY2,DYSQR,R,RSQR, 00002130
     1                  RSQRR,REYNLD,REY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H,00002140
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00002150
     3                 ,XNM1,UMNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6 00002160
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  00002170
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        00002180
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     00002190
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),    00002200
     3                  ETA(201),GA(201)                                00002210
      DIMENSION         IPO(64),IMO(64),INV(32),IAO(64),IAP(64)         00002220
      DOUBLE PRECISION  THETA,XNN                                       00002230
                                                                        00002240
      * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00002250
C                                                                       00002260
C     MISCELANEOUS PARAMETERS                                           00002270
C                                                                       00002280
      PI=3.1415926535898                                                00002290
      H=3.1415926535898                                                 00002300
C                                                                       00002310
      ALPHA=1.D0                                                        00002320
C                                                                       00002330
      H=PI/ALPHA                                                        00002340
      KINT=50                                                           00002350
      NITMAX=15                                                         00002360
      NP2=N+2                                                           00002370
      NP1=N+1                                                           00002380
      NM1=N-1                                                           00002390
      NM2=N-2                                                           00002400
      NM3=N-3                                                           00002410
      NM4=N-4                                                           00002420
      NH=N/2+1                                                          00002430
      MM(1)=M                                                           00002440
```

```
      MM(2)=0                                                           00002450
      MM(3)=0                                                           00002460
      M=2**M                                                            00002470
      MH=M/2                                                            00002480
      MHP=MH+1                                                          00002490
      MP1=M+1                                                           00002500
      MP2=M+2                                                           00002510
      MP3=M+3                                                           00002520
      M2M=M+M-1                                                         00002530
      MM1=M-1                                                           00002540
      XN=DFLOAT(N)                                                      00002550
      XNM1=DFLOAT(NM1)                                                  00002560
      XM=DFLOAT(M)                                                      00002570
      DX=2.DO*H/XM                                                      00002580
      DX2=2.DO*DX                                                       00002590
      DXSQR=DX*DX                                                       00002600
      DXSQR4=4.DO*DXSQR                                                 00002610
      DXSQR8=8.DO*DXSQR                                                 00002620
      DY=2.DO/XNM1                                                      00002630
      DY2=2.DO*DY                                                       00002640
      DYSQR=DY*DY                                                       00002650
      R=DX/DY                                                           00002660
      RSQR=R*R                                                          00002670
      RSQRP=RSQR+1.DO                                                   00002680
      REY=1.DO/REYNLD                                                   00002690
      TOL=.00002DO                                                      00002700
      ATERM=1.DO/(2.DO*DYSQR)                                           00002710
      BTERM=XM*DX2*DY2                                                  00002720
      FA=1.DO/(XM*XM)                                                   00002730
      FB=1.DO/(DY2**2)                                                  00002740
      FC=1.DO/(8.DO*(XM*DY)**2)                                         00002750
      THETA=2.DO*PI/XM                                                  00002760
      WRITE(6,100)R,RSQR,RSQRP                                          00002770
  100 FORMAT(8X,F12.8,3X,F12.8,3X,F12.8)                                00002780
C                                                                       00002790
C                                                                       00002800
C     SET UP PERIODIC COUNTERS                                          00002810
C     SET UP INDEXING COUNTERS, IAO &IAP.                               00002820
C                                                                       00002830
      JC=0                                                              00002840
      DO 10 I=1,M                                                       00002850
      IPO(I)=I+1                                                        00002860
      IMO(I)=I-1                                                        00002870
      IAO(I)=I+JC                                                       00002880
      IAP(I)=IAO(I)+1                                                   00002890
      JC=JC+1                                                           00002900
   10 CONTINUE                                                          00002910
      IPO(M)=1                                                          00002920
```

```
      IMO(1)=M                                                          00002930
C                                                                       00002940
CC    SET UP COORDINATES                                                00002950
C                                                                       00002960
      DO 20 J=1,N                                                       00002970
      Y(J)=DFLOAT(J-1)*DY-1.D0                                          00002980
   20 CONTINUE                                                          00002990
      DO 30 I=1,M                                                       00003000
      X(I)=DFLOAT(I-1)*DX-H+DX/2.D0                                     00003010
   30 CONTINUE                                                          00003020
C                                                                       00003030
CC    SET UP MODAL CONSTANTS. --- ALPHA & EE .                          00003040
C                                                                       00003050
      ALPHA=DYSQR/DXSQR                                                 00003060
      XALPHA(1)=2.D0                                                    00003070
      EE(1)=0.D0                                                        00003080
      DO 35 I=2,MH                                                      00003090
      IM=I-1                                                            00003100
      IA=IAP(IM)                                                        00003110
      IP=IA+1                                                           00003120
      XNN=DFLOAT(IM)                                                    00003130
      XALPHA(IA)=2.D0*(1.D0-ALPHA*(DCOS(THETA*XNN)-1.D0))               00003140
      XALPHA(IP)=XALPHA(IA)                                             00003150
      EE(I)=1.D0/DXSQR*(DSIN((2.D0*PI*XNN)/XM))**2                      00003160
   35 CONTINUE                                                          00003170
      XNN=DFLOAT(MH)                                                    00003180
      XALPHA(M)=2.D0*(1.D0-ALPHA*(DCOS(THETA*XNN)-1.D0))                00003190
C                                                                       00003200
CC    SET UP S AND INV TABLES FOR DHARM.                                00003210
C                                                                       00003220
      CALL DHARM(A,MM,INV,S,0,IERRP)                                    00003230
C                                                                       00003240
CC    LAMINAR FLOW STREAM FUNCTION AND VELOCITY                         00003250
C                                                                       00003260
      DO 40 J=2,NM1                                                     00003270
      PHILAM(J)=.5*Y(J)*(Y(J)**2-3.)                                    00003280
      U(J)=-1.5*(Y(J)**2-1.)                                            00003290
   40 CONTINUE                                                          00003300
      PHILAM(1)=1.                                                      00003310
      PHILAM(N)=-1.                                                     00003320
      U(1)=0.                                                           00003330
      U(N)=0.                                                           00003340
C                                                                       00003350
      RETURN                                                            00003360
      END                                                               00003370
```

```
      SUBROUTINE START                                                  00003380
      COMMON / FIELDS /  PHI,GAMA,EG,GAMOLD                             00003390
      COMMON / VECTOR /  T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,  00003400
     1                   U,PHILAM,EE,ETA,GA,X,Y                         00003410
      COMMON / ARRAYS /  IPO,IMO,INV,IAO,IAP,MM(3)                      00003420
      COMMON / PARAMS /  N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1,00003430
     1                   M2M,MH,MHP,NPROB,ICOUNT,ITEL,INT,KSTEP,ISTART, 00003440
     2                   ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP,00003450
     3                   KINT                                           00003460
      COMMON / FACTOR /  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00003470
     1                   RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,00003480
     2                   EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00003490
     3                  ,XNM1,UMNELD,UELD,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6 00003500
      COMMON / SYMBOL /  DOT,BLANK,XXX                                  00003510
      DOUBLE PRECISION   DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR,00003520
     1                   RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,00003530
     2                   EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00003540
     3                  ,XNM1,UMNELD,UELD,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6 00003550
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  00003560
      DOUBLE PRECISION   UMEAN(201),TEMP(201),TURBKE(201),U(201),       00003570
     1                   PHILAM(201),T(201),A(12),S(32),XALPHA(64),     00003580
     2                   PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),   00003590
     3                   ETA(201),GA(201)                               00003600
      DIMENSION          IPO(64),IMO(64),INV(32),IAO(64),IAP(64)        00003610
      DOUBLE PRECISION   PI,THETA,C,R,XL                                00003620
      DOUBLE PRECISION EPS                                              00003630
                                                                        00003640
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00003650
                                                                        00003660
C     BOUNDARY VALUES OF STREAM FUNCTION                                00003670
                                                                        00003680
      PI=3.14159265358979                                               00003690
      DO 10 I=1,M                                                       00003700
      PHI(I,1)=0.                                                       00003710
      PHI(I,N)=0.                                                       00003720
   10 CONTINUE                                                          00003730
                                                                        00003740
C     ESTABLISH INITIAL VALUES OF STREAM FUNCTION                       00003750
C     USE ARRAYS UMEAN AND TEMP FOR TEMPORARY STORAGE                   00003760
                                                                        00003770
      DO 60 J=1,NH                                                      00003780
      READ(5,11) PHMEAN(J),GAMEAN(J)                                    00003790
   11 FORMAT(2F10.0)                                                    00003800
   60 CONTINUE                                                          00003810
      NHM=NH-1                                                          00003820
      DO 65 J=1,NHM                                                     00003830
```

```
      JC=N-J+1                                                          00003840
      PHMEAN(JC)=PHMEAN(J)                                              00003850
      GAMEAN(JC)=GAMEAN(J)                                              00003860
   65 CONTINUE                                                          00003870
   12 FORMAT(10X,3(F12.6,3X))                                           00003880
      EPS=.1D0/DSQRT(2.D0)                                              00003890
      DO 71 J=1,N                                                       00003900
      C=PHMEAN(J)                                                       00003910
      B=GAMEAN(J)                                                       00003920
      DO 70 I=1,M                                                       00003930
      THETA=PI/H*X(I)                                                   00003940
      PHI(I,J)=EPS*(C*DCOS(THETA)+B*DSIN(THETA))                        00003950
   70 CONTINUE                                                          00003960
      WRITE(6,12) Y(J),C,B                                              00003970
   71 CONTINUE                                                          00003980
C                                                                       00003990
CC    COMPUTE INITIAL VALUES OF VORTICITY                               00004000
C                                                                       00004010
      DO 15 I=1,M                                                       00004020
      IP=IPO(I)                                                         00004030
      IM=IMO(I)                                                         00004040
      GAMA(I,N)=(8.D0*PHI(I,NM1)-PHI(I,NM2))*ATERM                      00004050
      GAMA(I,1)=(8.D0*PHI(I,2)-PHI(I,3))*ATERM                          00004060
      GAMOLD(I,1)=GAMA(I,1)                                             00004070
      GAMOLD(I,N)=GAMA(I,N)                                             00004080
      DO 15 J=2,NM1                                                     00004090
      GAMA(I,J)=(RSQR*(PHI(I,J+1)+PHI(I,J-1))+PHI(IP,J)+PHI(IM,J)       00004100
     1          -2.D0*RSQRP*PHI(I,J))/DXSQR                             00004110
      GAMOLD(I,J)=GAMA(I,J)                                             00004120
   15 CONTINUE                                                          00004130
      CALL MEANS                                                        00004140
      CALL TIMER                                                        00004150
      RETURN                                                            00004160
      END                                                               00004170

      SUBROUTINE TITLE                                                  00004180
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              00004190
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00004200
     1                  U,PHILAM,EE,ETA,GA,X,Y                          00004210
      COMMON / ARRAYS / IPO,IMO,INV,IAC,IAP,MM(3)                       00004220
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 00004230
     1                  M2M,MH,MHP,NPROB,ICOUNT,ITEL,INT,KSTEP,ISTART, 00004240
     2                  ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP,00004250
     3                  KINT                                            00004260
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR3,DY,DY2,DYSQR,R,RSQR, 00004270
     1                  RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,00004280
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00004290
```

```
     3                ,XNM1,UMNELD,UELD,TOL,FA,FB,FC,F1,F2,F3,F4,FF,F6  00004300
      COMMON / SYMBOL / DOT,BLANK,XXX                                    00004310
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR,  00004320
     1                  RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H, 00004330
     2                  EKMEAN,EKTURB,UVPAR,ATERM,BTERM,ALPHA,BETA,XM,XN 00004340
     3                 ,XNM1,UMNELD,UELD,TOL,FA,FB,FC,F1,F2,F3,F4,FF,F6  00004350
      DIMENSION    PHI(64,201),GAMA(64,201),FC(64,201),GAMOLD(64,201)    00004360
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),         00004370
     1                  PHILAM(201),T(201),A(128),S(32),XALPH(64),       00004380
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),FF(32),     00004390
     3                  ETA(201),GA(201)                                 00004400
      DIMENSION         IPO(64),IMO(64),INV(32),IAD(64),IAP(64)          00004410
C                                                                        00004420
CCC   * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *  00004430
C                                                                        00004440
      WRITE(6,11)                                                        00004450
      WRITE(6,22)                                                        00004460
      WRITE(6,12)NPROB                                                   00004470
      IF(NPROB .LT. 0) WRITE(6,23)                                       00004480
      WRITE(6,13)M,N                                                     00004490
      WRITE(6,14)REYNLD                                                  00004500
      WRITE(6,15)DX                                                      00004510
      WRITE(6,16)DY                                                      00004520
      WRITE(6,18) TOL                                                    00004530
      WRITE(6,19)XLAMDA                                                  00004540
      WRITE(6,21)FRACEL                                                  00004550
      HH=PI/H                                                            00004560
      WRITE(6,24) H,HH                                                   00004570
   24 FORMAT(1H ,10X,'H=',14X,F7.5,5X,'ALPHA=',3X,F7.5)                  00004580
   11 FORMAT(1H ,10X,'TURBCODE MARK VII   G. D. O''BRIEN'//)             00004590
   12 FORMAT(1H0,10X,'PROBLEM NUMBER',2X,I4)                             00004600
   13 FORMAT(1H0,10X,'MESH SIZE    ',I3,' BY ',I3)                       00004610
   14 FORMAT(1H ,10X,'REYNOLDS NUMBER=',F8.1)                            00004620
   15 FORMAT(1H ,10X,'DX=',13X,F7.6)                                     00004630
   16 FORMAT(1H ,10X,'DY=',13X,F7.6)                                     00004640
   18 FORMAT(1H ,10X,'TOL=',12X,F7.6)                                    00004650
   19 FORMAT(1H ,10X,'LAMDA=',10X,F7.5)                                  00004660
   21 FORMAT(1H ,10X,'FRACEL=',9X,F7.5)                                  00004670
   22 FORMAT(1H ,10X,'DUFORT-FRANKEL CENTRAL TIME DIFFERENCING')         00004680
   23 FORMAT(1H+,32X,'CONTINUED')                                        00004690
      RETURN                                                             00004700
      END                                                                00004710
```

```
      SUBROUTINE STEP                                                   00004720
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              00004730
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00004740
     1                  U,PHILAM,EE,ETA,GA,X,Y                         00004750
      COMMON / ARRAYS / IPO,IMO,INV,IAO,IAP,MM(3)                       00004760
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 00004770
     1                  M2M,MH,MHP,NPSOR,ICOUNT,ITEL,INT,KOUTP,ISTART, 00004780
     2                  ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP,00004790
     3                  KINT                                           00004800
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR2,DY,DY2,DYSQR,R,RSQR, 00004810
     1                  RSQRP,REYNLD,PEY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H,00004820
     2                  EKMEAN,EKTURB,UVRAP,ATERM,BTERM,ALPHA,BETA,XM,XN00004830
     3                 ,XNM1,UMNELO,UFLO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6 00004840
      COMMON / SYMBOL / DOT,BLANK,XXX                                   00004850
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR2,DY,DY2,DYSQR,R,RSQR, 00004860
     1                  RSQRP,REYNLD,PEY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H,00004870
     2                  EKMEAN,EKTURB,UVRAP,ATERM,BTERM,ALPHA,BETA,XM,XN00004880
     3                 ,XNM1,UMNELO,UFLO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6 00004890
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  00004900
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        00004910
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     00004920
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),    00004930
     3                  ETA(201),GA(201)                                00004940
      DIMENSION         IPO(64),IMO(64),INV(32),IAO(64),IAP(64)         00004950
      DOUBLE PRECISION  GJ,GJ1,GJ2,GJ3,Q                                00004960
      DOUBLE PRECISION  CRIT,TST,PUNT,GAMAIJ                            00004970
                                                                        00004980
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00004990
C                                                                       00005000
      REAL*8  LAMDA,LAMDAB,LAMDAC,LAMDAD,LAMDAE                         00005010
C                                                                       00005020
C     THIS ROUTINE COMPUTES NEW VALUES OF VORTICITY AT TIME+DT BY       00005030
C     MODIFIED EULER IMPLICIT TIME DIFFERENCING.                        00005040
C     ARRAY EG IS USED TO STORE TERMS EVALUATED AT TIME T.              00005050
C                                                                       00005060
      IF(KINT .EQ. 1) GO TO 6                                           00005070
      IF( ICOUNT .EQ. 0) GO TO 6                                        00005080
      DO 5 I=1,M                                                        00005090
      DO 5 J=1,N                                                        00005100
      GAMA(I,J)=.5*(GAMA(I,J)+GAMOLD(I,J))                              00005110
      GAMOLD(I,J)=GAMA(I,J)                                             00005120
    5 CONTINUE                                                          00005130
      CALL PRESUR                                                       00005140
      TIME=TIME-DT/2.D0                                                 00005150
    6 CALL TIMER                                                        00005160
      LAMDA=RSQRP*DT/(REYNLD*DXSQR)                                     00005170
```

```
      LAMDAB=(1.D0-LAMDA)/(1.D0+LAMDA)                                  00005180
      LAMDAC=DT/(2.D0*(1.D0+LAMDA)*DXSQR)                               00005190
      LAMDAD=DT/(4.D0*(1.D0+LAMDA)*DX)                                  00005200
      LAMDAE=3.D0*LAMDAD                                                00005210
      DO 10 I=1,M                                                       00005220
      IP=IPO(I)                                                         00005230
      IM=IMO(I)                                                         00005240
      DO 10 J=2,NM1                                                     00005250
      JP=J+1                                                            00005260
      JM=J-1                                                            00005270
      GJ1= (PHI(IM,J)-PHI(IP,J))*(GAMA(I,JP)-GAMA(I,JM))               00005280
     1     +(PHI(I,JP)-PHI(I,JM))*(GAMA(IP,J)-GAMA(IM,J))              00005290
      GJ2= GAMA(IP,JP)*(PHI(I,JP)-PHI(IP,J))                           00005300
     1     +GAMA(IM,JM)*(PHI(I,JM)-PHI(IM,J))                          00005310
     2     +GAMA(IP,JM)*(PHI(IP,J)-PHI(I,JM))                          00005320
     3     +GAMA(IM,JP)*(PHI(IM,J)-PHI(I,JP))                          00005330
      GJ3= GAMA(I,JP)*(PHI(IM,JP)-PHI(IP,JP))                          00005340
     1     +GAMA(IM,J)*(PHI(IM,JM)-PHI(IM,JP))                         00005350
     2     +GAMA(I,JM)*(PHI(IP,JM)-PHI(IM,JM))                         00005360
     3     +GAMA(IP,J)*(PHI(IP,JP)-PHI(IP,JM))                         00005370
      GJ=(GJ1+GJ2+GJ3)/12.D0                                           00005380
      GJ=R*GJ                                                          00005390
      Q=RSQR*(GAMA(I,JP)+GAMA(I,JM))+GAMA(IP,J)+GAMA(IM,J)             00005400
      Q=Q*PEY                                                          00005410
      FG(I,J)=LAMDAB*GAMA(I,J)+LAMDAC*(GJ+Q)                           00005420
     1          -LAMDAD*U(J)*(GAMA(IP,J)-GAMA(IM,J))                   00005430
     2          -LAMDAE*(PHI(IP,J)-PHI(IM,J))                          00005440
   10 CONTINUE                                                         00005450
      NITSUM=0                                                         00005460
C                                                                      00005470
C     FIRST SWEEP. START Y PASS AT LOWER WALL                          00005480
C                                                                      00005490
      NITER=0                                                          00005500
   15 PUNT=0.                                                          00005510
      DO 20 I=1,M                                                      00005520
      IP=IPO(I)                                                        00005530
      IM=IMO(I)                                                        00005540
      DO 20 J=2,NM1                                                    00005550
      JP=J+1                                                           00005560
      JM=J-1                                                           00005570
      GJ1= (PHI(IM,J)-PHI(IP,J))*(GAMA(I,JP)-GAMA(I,JM))               00005580
     1     +(PHI(I,JP)-PHI(I,JM))*(GAMA(IP,J)-GAMA(IM,J))              00005590
      GJ2= GAMA(IP,JP)*(PHI(I,JP)-PHI(IP,J))                           00005600
     1     +GAMA(IM,JM)*(PHI(I,JM)-PHI(IM,J))                          00005610
     2     +GAMA(IP,JM)*(PHI(IP,J)-PHI(I,JM))                          00005620
     3     +GAMA(IM,JP)*(PHI(IM,J)-PHI(I,JP))                          00005630
      GJ3= GAMA(I,JP)*(PHI(IM,JP)-PHI(IP,JP))                          00005640
     1     +GAMA(IM,J)*(PHI(IM,JM)-PHI(IM,JP))                         00005650
```

```
     2     +GAMA(I,JM)*(PHI(IP,JM)-PHI(IM,JM))                                00005660
     3     +GAMA(IP,J)*(PHI(IP,JP)-PHI(IP,JM))                                00005670
      GJ=(GJ1+GJ2+GJ3)/12.D0                                                  00005680
      GJ=R*GJ                                                                 00005690
      Q=RSQR*(GAMA(I,JP)+GAMA(I,JM))+GAMA(IP,J)+GAMA(IM,J)                    00005700
      Q=Q*REY                                                                 00005710
      GAMA(I,J)=LAMDAC*(GJ+Q)-LAMDAD*U(J)*(GAMA(IP,J)-GAMA(IM,J))             00005720
     1          -LAMDAE*(PHI(IP,J)-PHI(IM,J))+EG(I,J)                        00005730
   20 CONTINUE                                                                00005740
      CALL PRESUR                                                             00005750
C                                                                             00005760
CC    SECOND SWEEP. START Y PASS AT UPPER WALL                                00005770
C                                                                             00005780
      DO 30 I=1,M                                                             00005790
      IP=IPO(I)                                                               00005800
      IM=IMO(I)                                                               00005810
      J=NM1                                                                   00005820
      DO 30 JC=2,NM1                                                          00005830
      JP=J+1                                                                  00005840
      JM=J-1                                                                  00005850
      GJ1= (PHI(IM,J)-PHI(IP,J))*(GAMA(I,JP)-GAMA(I,JM))                     00005860
     1     +(PHI(I,JP)-PHI(I,JM))*(GAMA(IP,J)-GAMA(IM,J))                   00005870
      GJ2= GAMA(IP,JP)*(PHI(I,JP)-PHI(IP,J))                                 00005880
     1     +GAMA(IM,JM)*(PHI(I,JM)-PHI(IM,J))                               00005890
     2     +GAMA(IP,JM)*(PHI(IP,J)-PHI(I,JM))                               00005900
     3     +GAMA(IM,JP)*(PHI(IM,J)-PHI(I,JP))                               00005910
      GJ3= GAMA(I,JP)*(PHI(IM,JP)-PHI(IP,JP))                                00005920
     1     +GAMA(IM,J)*(PHI(IM,JM)-PHI(IM,JP))                              00005930
     2     +GAMA(I,JM)*(PHI(IP,JM)-PHI(IM,JM))                              00005940
     3     +GAMA(IP,J)*(PHI(IP,JP)-PHI(IP,JM))                              00005950
      GJ=(GJ1+GJ2+GJ3)/12.D0                                                  00005960
      GJ=R*GJ                                                                 00005970
      Q=RSQR*(GAMA(I,JP)+GAMA(I,JM))+GAMA(IP,J)+GAMA(IM,J)                    00005980
      Q=Q*REY                                                                 00005990
      TST=GAMA(I,J)                                                           00006000
      GAMAIJ=    LAMDAC*(GJ+Q)-LAMDAD*U(J)*(GAMA(IP,J)-GAMA(IM,J))           00006010
     1          -LAMDAE*(PHI(IP,J)-PHI(IM,J))+EG(I,J)                        00006020
C                                                                             00006030
CC    TEST FOR CONVERGENCE                                                    00006040
C                                                                             00006050
      CRIT=DABS(1.-GAMA(I,J)/GAMAIJ)                                          00006060
      PUNT=PUNT+CRIT                                                          00006070
   29 GAMA(I,J)=GAMAIJ                                                        00006080
      J=J-1                                                                   00006090
   30 CONTINUE                                                                00006100
      NITER=NITER+2                                                           00006110
      TST=PUNT/(XM*NM2)                                                       00006120
      IF(TST .LT. TOL) GO TO 31                                               00006130
```

```
      IF(NITER .GT. NITMAX) GO TO 31                                    00006140
      CALL PRESUR                                                       00006150
      GO TO 15                                                          00006160
   31 TIME=TIME+DT                                                      00006170
      NITOT=NITOT+NITER                                                 00006180
C     IF ICOUNT EQUALS KSTEP THEN SUBROUTINE SPCTRM WILL BE BYPASSED    00006190
C     SO THAT  NO SPECTRA WILL BE CALCULATED DURING THE ITERATIONS      00006200
C     IN SUBROUTINE STEP                                                00006210
      KSTEP=KSTEP+1                                                     00006220
      CALL PRESUR                                                       00006230
      KSTEP=KSTEP-1                                                     00006240
   40 RETURN                                                            00006250
      END                                                               00006260
```

```
      SUBROUTINE TIMER                                                  00006270
      COMMON / FIELDS / PHI,GAMA,FG,GAMOLD                              00006280
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00006290
     1                  U,PHILAM,FF,ETA,GA,X,Y                         00006300
      COMMON / ARRAYS / IPO,IMO,INV,IAO,IAP,MM(3)                       00006310
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 00006320
     1                  M2M,MH,MHP,NPRCB,ICOUNT,ITEL,INT,KSTEP,ISTART, 00006330
     2                  ISCIP,VSCIP,ITAPE,NITER,NITOT,NITMAX,IEIG,MCOMP,00006340
     3                  KINT                                           00006350
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00006360
     1                  RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,00006370
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00006380
     3                 ,XNM1,UMNFLC,UFLO,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6 00006390
      COMMON / SYMBOL / DOT,BLANK,XXX                                   00006400
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00006410
     1                  RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,00006420
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00006430
     3                 ,XNM1,UMNFLC,UFLO,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6 00006440
      DIMENSION     PHI(64,201),GAMA(64,201),FG(64,201),GAMOLD(64,201)  00006450
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        00006460
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     00006470
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),FF(32),    00006480
     3                  ETA(201),GA(201)                                00006490
      DIMENSION         IPO(64),IMO(64),INV(32),IAO(64),IAP(64)         00006500
                                                                        00006510
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00006520
C                                                                       00006530
C     THIS ROUTINE COMPUTES A NEW VALUE OF DT.                          00006540
C                                                                       00006550
      UMAX=0.                                                           00006560
      VMAX=0.                                                           00006570
      DO 10 I=1,M,4                                                     00006580
      IP=IPO(I)                                                         00006590
      IM=IMO(I)                                                         00006600
      DO 10 J=1,NH,2                                                    00006610
      FLIP=(PHI(I,J)-PHI(I,J+2))/DY2+U(J+1)                             00006620
      FLIPR=(PHI(IP,J)-PHI(IM,J))/DX2                                   00006630
      UMAX=AMAX1(UMAX,ABS(FLIP))                                        00006640
      VMAX=AMAX1(VMAX,ABS(FLIPR))                                       00006650
   10 CONTINUE                                                          00006660
      DTT=FRACEL*DX/UMAX                                                00006670
      DT=FRACEL*DY/VMAX                                                 00006680
      DT=DMIN1(DT,DTT)                                                  00006690
      ITEL=1                                                            00006700
      IF(DT .EQ. DTT)ITEL=-1                                            00006710
      DT=FRACEL*DX*DY/(UMAX*DY+VMAX*DX)                                 00006720
```

```
      DTT=XLAMDA*DX*DY*REYNLD/4.DO                                      CCC06721
      DT=DMIN1(DT,DTT)                                                  CCC06722
      IF(DT .EQ. DTT) ITEL=0                                            CCC06723
      RETURN                                                            CCC06730
      END                                                               CCC06740
```

```
      SUBROUTINE ADVANC                                                 00006750
      COMMON / FIELDS / PHI,GAMA,FG,GAMOLD                              00006760
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00006770
     1                  U,PHILAM,FF,FTA,GA,X,Y                         00006780
      COMMON / ARRAYS / IPO,IMO,INV,IAC,IAP,MM(3)                       00006790
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 00006800
     1                  M2M,MH,VHD,NSCOR,ICOUNT,ITEL,INT,KSTEP,ISTART, 00006810
     2                  ISCID,MSCID,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP, 00006820
     3                  KINT                                           00006830
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00006840
     1                  RSQRP,REYNLD,REY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H, 00006850
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN 00006860
     3                 ,XNM1,UMNELD,UELD,TOL,EA,EB,EC,E1,E2,E3,E4,EE,E6  00006870
      COMMON / SYMBOL / DOT,BLANK,XXX                                   00006880
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00006890
     1                  RSQRP,REYNLD,REY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H, 00006900
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN 00006910
     3                 ,XNM1,UMNELD,UELD,TOL,EA,EB,EC,E1,E2,E3,E4,EE,E6  00006920
      DIMENSION     PHI(64,201),GAMA(64,201),FG(64,201),GAMOLD(64,201)  00006930
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        00006940
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     00006950
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),FF(32),    00006960
     3                  ETA(201),GA(201)                                00006970
      DIMENSION         IPO(64),IMO(64),INV(32),IAC(64),IAP(64)         00006980
      DOUBLE PRECISION  GJ1,GJ2,GJ3,GJ,0                               00006990
      REAL*8 LAMDA,LAMDAB,LAMDAC,LAMDAD,LAMDAE,LAMDAF,LAMDAG           00007000
C                                                                       00007010
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00007020
C                                                                       00007030
C                                                                       00007040
C     THIS ROUTINE GENERATES INTERIOR VALUES OF VORTICITY AT TIME=T+DT   00007050
C     BY SECOND ORDER CENTRAL TIME DIFFERENCING.                        00007060
C     GJ1,GJ2,GJ3 ARE EACH FINITE DIFFERENCE ANALOGS FOR THE CONVECTIVE 00007070
C     TERMS . Q IS THE DISSIPATION TERM                                 00007080
C     ARRAY FG IS USED FOR TEMPORARY STORAGE OF THE NEW VALUES OF       00007090
C     ARRAY GAMOLD STORES OLD VALUES OF THE VORTICITY FOR THE CENTRAL   00007100
C     VORTICITY                                                         00007110
C     TIME DIFFERENCING.                                                00007120
C                                                                       00007130
C                                                                       00007140
      IF(ICOUNT .EQ. KSTEP)GO TO 30                                     00007150
C                                                                       00007160
    1 LAMDA=2.DO*DT*RSQRP/(REYNLD*DXSQR)                                00007170
      LAMDAB=(1.-LAMDA)/(1.+LAMDA)                                      00007180
      LAMDAC=2.*DT/(DXSQR*(1.+LAMDA))                                   00007190
      LAMDAD=DT/(DX*(1.DO+LAMDA))                                       00007200
```

```
      LAMDAE=2.D0*LAMDAD                                                00007210
      DO 10 I=1,M                                                       00007220
      IP=IPO(I)                                                         00007230
      IM=IMO(I)                                                         00007240
      IMM=IMO(IM)                                                       00007250
      IPP=IPO(IP)                                                       00007260
      DO 10 J=2,NM1                                                     00007270
      JP=J+1                                                            00007280
      JM=J-1                                                            00007290
      GJ1= (PHI(IM,J)-PHI(IP,J))*(GAMA(I,JP)-GAMA(I,JM))               00007300
     1    +(PHI(I,JP)-PHI(I,JM))*(GAMA(IP,J)-GAMA(IM,J))               00007310
      GJ2= GAMA(IP,JP)*(PHI(I,JP)-PHI(IP,J))                           00007320
     1    +GAMA(IM,JM)*(PHI(I,JM)-PHI(IM,J))                           00007330
     2    +GAMA(IP,JM)*(PHI(IP,J)-PHI(I,JM))                           00007340
     3    +GAMA(IM,JP)*(PHI(IM,J)-PHI(I,JP))                           00007350
      GJ3= GAMA(I,JP)*(PHI(IM,JP)-PHI(IP,JP))                          00007360
     1    +GAMA(IM,J)*(PHI(IM,JM)-PHI(IM,JP))                          00007370
     2    +GAMA(I,JM)*(PHI(IP,JM)-PHI(IM,JM))                          00007380
     3    +GAMA(IP,J)*(PHI(IP,JP)-PHI(IP,JM))                          00007390
      GJ=(GJ1+GJ2+GJ3)/12.D0                                           00007400
      GJ=R*GJ                                                          00007410
      Q=RSOR*(GAMA(I,JP)+GAMA(I,JM))+GAMA(IP,J)+GAMA(IM,J)             00007420
      Q=Q*REY                                                          00007430
      FG(I,J)=LAMDAB*GAMOLD(I,J)+LAMDAC*(GJ+Q)                         00007440
     1        -LAMDAD*U(J)*(GAMA(IP,J)-GAMA(IM,J))                     00007450
     2        -LAMDAE*(PHI(IP,J)-PHI(IM,J))                            00007460
   10 CONTINUE                                                         00007470
      DO 20 I=1,M                                                      00007480
      DO 20 J=2,NM1                                                    00007490
      GAMOLD(I,J)=GAMA(I,J)                                            00007500
      GAMA(I,J)=FG(I,J)                                                00007510
   20 CONTINUE                                                         00007520
      CALL PRESUP                                                      00007530
      TIME=TIME+DT                                                     00007540
      NITER=0                                                          00007550
      GO TO 40                                                         00007560
C                                                                      00007570
   30 CALL STEP                                                        00007580
      KSTEP=KSTEP+KINT                                                 00007590
   40 RETURN                                                           00007600
      END                                                              00007610
```

```
      SUBROUTINE PRESUR                                                 00007620
      COMMON / FIELDS /  PHI,GAMA,EG,GAMOLD                             00007630
      COMMON / VECTOR /  T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,  00007640
     1                   U,PHILAM,EE,ET4,GA,X,Y                         00007650
      COMMON / ARRAYS /  IPO,IMO,INV,IAO,IAP,MM(3)                      00007660
      COMMON / PARAMS /  N,M,MP1,MP2,MM1,MM2,MM3,MM4,NH,MP1,MP2,MP3,MM1,00007670
     1                   M2M,MH,MHP,MPLOP,ICOUNT,ITEL,INT,KSTEP,ISTART,00007680
     2                   ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP,00007690
     3                   KINT                                           00007700
      COMMON / FACTOR /  DX,DX2,DXSQR,DXSQR4,DXSQR2,DY,DY2,DYSQR,R,RSQR,00007710
     1                   RSQRP,REYNLD,PEY,FPACEL,XLAMDA,TIME,DT,DTT,PI,H,00007720
     2                   EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00007730
     3                  ,XNM1,UMNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,EE,E6 00007740
      DOUBLE PRECISION   DX,DX2,DXSQR,DXSQR4,DXSQR2,DY,DY2,DYSQR,R,RSQR,00007750
     1                   RSQRP,REYNLD,PEY,FPACEL,XLAMDA,TIME,DT,DTT,PI,H,00007760
     2                   EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN00007770
     3                  ,XNM1,UMNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,EE,E6 00007780
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  00007790
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        00007800
     1                   PHILAM(201),T(201),A(128),S(32),XALPHA(64),    00007810
     2                   PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),   00007820
     3                   ET4(201),GA(201)                               00007830
      DIMENSION          IPO(64),IMO(64),INV(32),IAO(64),IAP(64)        00007840
C                                                                       00007850
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00007860
C                                                                       00007870
C     THIS ROUTINE SOLVES A POISSON EQUATION FOR THE STREAM FUNCTION    00007880
C     BY MEANS OF THE DISCRETE FOURIER TRANSFORM.                       00007890
C                                                                       00007900
C      OBTAIN THE FOURIER TRANSFORM OF GAMA. STORE IN ARRAY PHI.        00007910
C                                                                       00007920
      DO 20 J=1,N                                                       00007930
      DO 10 I=1,M                                                       00007940
      IA=IAO(I)                                                         00007950
      IP=IAP(I)                                                         00007960
      A(IA)=GAMA(I,J)                                                   00007970
      A(IP)=0.                                                          00007980
   10 CONTINUE                                                          00007990
C                                                                       00008000
      CALL DHARM(A,MM,INV,S,2,IERRP)                                    00008010
      PHI(1,J)=A(1)                                                     00008020
      DO 20 I=2,M                                                       00008030
      PHI(I,J)=A(I+1)                                                   00008040
   20 CONTINUE                                                          00008050
C     OBTAIN THE MEAN VALUE OF GAMA FROM THE ZERO MODE                  00008060
      DO 25 J=1,N                                                       00008070
```

```
      GAMEAN(J)=PHI(1,J)/XM+3.D0*Y(J)                                   00008080
   25 CONTINUE                                                          00008090
C     OBTAIN THE SPECTRUM OF SQUARED VORTICITY                          00008100
      L=ICOUNT+1                                                        00008110
      IF(L .EQ. INT) CALL SPCTRM(-1)                                    00008120
C                                                                       00008130
CC    SOLVE FOR THE FOURIER TRANSFORM OF PHI. STORE IN ARRAY PHI.       00008140
C                                                                       00008150
      DO 40 I=1,M                                                       00008160
      PHI(I,N)=0.                                                       00008170
      PHI(I,1)=0.                                                       00008180
      ALPHA=XALPHA(I)                                                   00008190
      DO 30 J=1,NM2                                                     00008200
      T(J)=PHI(I,J+1)                                                   00008210
   30 CONTINUE                                                          00008220
      CALL TRISOL                                                       00008230
      DO 40 J=2,NM1                                                     00008240
      PHI(I,J)=T(J-1)                                                   00008250
   40 CONTINUE                                                          00008260
C     OBTAIN THE MEAN VALUE OF PHI FROM THE ZERO MODE                   00008270
      DO 45 J=2,NM1                                                     00008280
      PHMEAN(J)=PHI(1,J)/XM                                             00008290
   45 CONTINUE                                                          00008300
      PHMEAN(N)=0.D0                                                    00008310
      PHMEAN(1)=0.D0                                                    00008320
C     OBTAIN THE ENERGY SPECTRUM.                                       00008330
      IF(L .EQ. INT) CALL SPCTRM(1)                                     00008340
C                                                                       00008350
C                                                                       00008360
      IF(L .EQ. INT) GO TO 1000                                         00008370
      IF((ICOUNT .NE. ISTART) .AND. (ICOUNT .NE. NCYCLS)) GO TO 1004    00008380
 1000 CONTINUE                                                          00008390
C     PRINT NORMALIZED MODE SHAPES                                      00008400
      CALL MODES(1,2,1)                                                 00008410
      CALL MODES(1,2,2)                                                 00008420
      CALL MODES(3,4,1)                                                 00008430
      CALL MODES(3,4,2)                                                 00008440
C                                                                       00008450
 1004 CONTINUE                                                          00008460
C                                                                       00008470
C     INVERT THE FOURIER TRANSFORM OF PHI                               00008480
C                                                                       00008490
      DO 70 J=2,NM1                                                     00008500
      A(1)=PHI(1,J)                                                     00008510
      A(2)=0.                                                           00008520
      DO 50 I=3,MP1                                                     00008530
      A(I)=PHI(I-1,J)                                                   00008540
   50 CONTINUE                                                          00008550
```

```
      A(MP2)=0.                                                         00008560
      IC=5                                                              00008570
      DO 60 I=MP3,M2M,2                                                 00008580
      A(I)=PHI(I-IC,J)                                                  00008590
      IP=I+1                                                            00008600
      A(IP)=-PHI(IP-IC,J)                                               00008610
      IC=IC+4                                                           00008620
   60 CONTINUE                                                          00008630
      CALL DHARM(A,MM,INV,S,-2,IERR)                                    00008640
      DO 70 I=1,M                                                       00008650
      IA=IAD(I)                                                         00008660
      PHI(I,J)=A(IA)                                                    00008670
   70 CONTINUE                                                          00008680
   11 FORMAT(10X,'**  SET OF GAMA  **'//)                              00008690
   12 FORMAT(10X,'**  SOLUTION FOR SET OF PHI  **'//)                  00008700
   13 FORMAT(10X,'**  SOLUTION FOR PHI  **'//)                         00008710
   14 FORMAT(10X,'**  SET OF EXACT SOLUTION FOR PHI  **'//)            00008720
C                                                                       00008730
C     COMPUTE NEW VALUES OF VORTICITY AT WALLS                          00008740
C                                                                       00008750
      DO 80 I=1,M                                                       00008760
      GAMA(I,N)=(8.D0*PHI(I,NM1)-PHI(I,NM2))*ATERM                      00008770
      GAMA(I,1)=(8.D0*PHI(I,2)-PHI(I,3))*ATERM                          00008780
   80 CONTINUE                                                          00008790
      RETURN                                                            00008800
      END                                                               00008810
```

```
      SUBROUTINE TRISOL                                                 C0000820
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              C0000830
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   C0000840
     1                  U,PHILAM,FF,ETA,GA,X,Y                          C0000850
      COMMON / ARRAYS / IPO,IMO,INV,IAC,IAP,MM(3)                       C0000860
      COMMON / PARAMS / N,M,NP1,NP2,MM1,MM2,MM3,MM4,MH,MP1,MP2,MP3,MM1, C0000870
     1                  M2M,MH,MHP,NPSCR,ICOUNT,ITEL,INT,NSTEP,ISTART, C0000880
     2                  ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,NCOMP,C0000890
     3                  KINT                                            C0000900
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, C0000910
     1                  RSQR8,REYNLD,DEY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H, C0000920
     2                  EKMEAN,EKTURB,UVRAD,ATERM,BTERM,ALPHA,BETA,XM,XN C0000930
     3                 ,XNM1,UNNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6  C0000940
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, C0000950
     1                  RSQR8,REYNLD,DEY,ERACEL,XLAMDA,TIME,DT,DTT,PI,H, C0000960
     2                  EKMEAN,EKTURB,UVRAD,ATERM,BTERM,ALPHA,BETA,XM,XN C0000970
     3                 ,XNM1,UNNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6  C0000980
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  C0000990
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        C0001000
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     C0001010
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),FF(32),    C0001020
     3                  ETA(201),GA(201)                                C0001030
      DIMENSION         IPO(64),IMO(64),INV(32),IAC(64),IAP(64)         C0001040
C                                                                       C0001050
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * C0001060
C                                                                       C0001070
C     THIS SUBROUTINE SOLVES THE SYSTEM OF EQUATIONS P*X=DYSQR*Y.       C0001080
C     X AND Y ARE VECTORS,DYSQR IS A SCALAR, P IS A TRIDIAGONAL MATRIX  C0001090
C     WITH ALL MAIN DIAGONAL ELEMENTS=-ALPHA,UPPER AND LOWER DIAGONAL   C0001100
C     ELEMENTS=UNITY. ON INPUT Y IS STORED IN ARRAY T.  ON OUTPUT T     C0001110
C     CONTAINS THE SOLUTION VECTOR X. ALL VECTORS HAVE DIMENSION NM2    C0001120
C                                                                       C0001130
C                                                                       C0001140
      GA(1)=-ALPHA                                                      C0001150
      ETA(1)= T(1)/GA(1)                                                C0001160
      DO 10 J=2,NM2                                                     C0001170
      GA(J) = - ALPHA -1./GA(J-1)                                       C0001180
      ETA(J) = (T(J)-ETA(J-1))/GA(J)                                    C0001190
   10 CONTINUE                                                          C0001200
      T(NM2)=DYSQR*ETA(NM2)                                             C0001210
      J = NM2-1                                                         C0001220
      DO 20 JC=2,NM2                                                    C0001230
      T(J)=DYSQR*ETA(J)-T(J+1)/GA(J)                                    C0001240
      J=J-1                                                             C0001250
   20 CONTINUE                                                          C0001260
      RETURN                                                            C0001270
```

00005280

END

```
      SUBROUTINE MODES(IMOD1,IMOD2,ISTYLE)
      COMMON / FIELDS /  PHI,GAMMA,EG,GAMOLD
     1COMMON / VECTOR /  U,PHILAM,ETA,GA,X,V
      COMMON / ARRAYS /  IBC,IMO,INV,IAD,MM(3)
      COMMON / PARAMS /  N,M,NP1,ND2,NM1,NM2,NM3,NM4,NM,MP1,M22,MP3,MM1,
     1M2M,MM,MHP,NBCB,ICOUNT,ITEL,INT,KSTEP,ISTAD,
     2ISTD,MSTD,TABS,NTEP,NTST,NITMAX,TEIG,MCMD,
     3KINT
      COMMON / FACTOR /  DX,DX2,DXSQR,DXSQR4,DXSQR6,DY,DY2,DYSQR,R,RSQR,
     1RSQR4,REYNLD,REY,EPSACC,XLAMDA,TIME,DT,ALPHA,BETA,XM,XN,
     2EKMEAN,EKTURB,UVPAR,ATERM,EC,EI,E2,ER,E4,E5,
     3XNU,TIMEL,TOL,E1,EP,
      COMMON / SYMBOL /  DCOT,RLANK,XXX
     DOUBLE PRECISION    DX,DX2,DXSQR,DXSQR4,DXSQR6,DY,DY2,DYSQR,R,RSQR,
     1RSQR4,REY,VMIL,
     2EKMEAN,EKTURB,UVPAR,ATERM,ALPHA,BETA,XM,XN,
     3XNU,TIMEL,TOL,EP,EC,E1,E2,ER,E4,E5,
     4DCOT,RLANK,XXX
      DIMENSION          PHI(64,20),GAMA(64,20),TEE(20),TURBKE(20),
     1UMEAN(20),TE(20),CAMCL(20),U(20,20),
     2DHTLAM(20),T(20),S(32),XALPHA(34),
     3PHMEAN(20),A(32),
     4ETA(20),X(64),Y(20),EE(32),
     5EDD(64),TMU(64),INV(32),IAD(64),IAP(64)
     DOUBLE PRECISION    UMEAN,TE,TEE,GAMA,PHI,TURBKE,CAMCL,PHMEAN,ETA,
     1DHTLAM,T,S,XALPHA,A,EDD,TMU,INV,IAD,IAP
C
C  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C  THIS ROUTINE COMPUTES AND PRINTS NORMALIZED MODAL FUNCTIONS
C    OF THE DISTURBANCE STREAM FUNCTION.
C
      IM1=IAP(IMOD1)
      IM1P=IM1+1
      IM2=IAP(IMOD2)
      IM2P=IM2+1
      IF(ISTYLE.EQ.2) GO TO 20
      A1=PHI((IM1-1)*NP1+1,NH)**2+PHI((IM1P-1)*NP1+1,NH)**2
      A2=PHI((IM2-1)*NP1+1,NH)**2+PHI((IM2P-1)*NP1+1,NH)**2
      A3=-PHI((IM2P-1)*NP1+1,NH)/A3
      WRITE(6,11) IMOD1,IMOD2
   10 FORMAT(1H0,5X,1HJ,5X,1HN,5X,4HETA(1),5X,4HETA(2),5X,4HETA(3))
      DO 15 J=1,N
      ETA(1)=A1*PHI((IM1-1)*NP1+J)-A1*PHI((IM1P-1)*NP1+J)
      ETA(2)=A1*PHI((IM1-1)*NP1+J)+A2*PHI((IM1P-1)*NP1+J)
      ETA(3)=A4*PHI((IM2-1)*NP1+J)-A3*PHI((IM2P-1)*NP1+J)
```

```
      ETA(4)=A3*PHI(IM2,J)+A4*PHI(IM2P,J)                               00009750
      WRITE(6,12) J,ETA(1),ETA(2),Y(J),ETA(3),ETA(4),J                  00009760
   10 CONTINUE                                                          00009770
   11 FORMAT(1HO,5X,'J',4X,'REAL    MODE ',I3,'  IMAG',11X,'Y',9X,      00009780
     1       'REAL    MODE ',I3,'  IMAG',6X,'J')                       00009790
   12 FORMAT(4X,I3,2(3X,E10.6),6X,E7.4,6X,2(E10.6,3X),I3)               00009800
      GO TO 40                                                          00009810
   20 WRITE(6,21) IMOD1,IMOD2                                           00009820
      DO 30 J=1,N                                                       00009830
      ETA(1)=PHI(IM1,J)**2+PHI(IM1P,J)**2                               00009840
      ETA(1)=DSQRT(ETA(1))/XM                                           00009850
      ETA(2)=ATAN2(PHI(IM1,J),PHI(IM1P,J))                              00009860
      ETA(3)=PHI(IM2,J)**2+PHI(IM2P,J)**2                               00009870
      ETA(3)=DSQRT(ETA(3))/XM                                           00009880
      ETA(4)=ATAN2(PHI(IM2,J),PHI(IM2P,J))                              00009890
      WRITE(6,12) J,ETA(1),ETA(2),Y(J),ETA(3),ETA(4),J                  00009900
   30 CONTINUE                                                          00009910
   21 FORMAT(1HO,5X,'J',4X,'MAG     MODE ',I3,'  PHAS',11X,'Y',9X,      00009920
     1       'MAG     MODE ',I3,'  PHAS',6X,'J')                       00009930
   40 IFIG=1                                                            00009940
      RETURN                                                            00009950
      END                                                               00009960
```

```
      SUBROUTINE SPCTRM(INDEX)
      COMMON / FIELDS /  PHI,GAMA,EG,CAMOLD
      COMMON / VECTOR /  T,A,S,XALPHA,BETAMEAN,GAMEAN,UMEAN,TEMP,TURBKE,
     1 COMMON / ARRAYS /  U,PHILAM,ENV,TAB,MM(3)
      COMMON / PARAMS /  N,M,NP1,ND2,NM1,NM2,NM3,NM4,MP1,MP2,MP3,MM1,
     1           M2M,MH,MHP,NDP,NDP2,IHT,ITEL,INT,KC,IST,ART,
     2           ISCID,MSCID,NITER,NITMAX,TFIC,MCOMP,
     3           KTNT
      COMMON / FACTOR /  DX,DX2,DXSQR,DXSQR4,DXSQR2,DY,DY2,DYSQR,R,RSQR,
     1           RSQRP,REYNLD,DEY,EGACEL,XLAMCL,TIME,DT,DTT,PI,XN,
     2           EKMEAN,EKTUB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,E2,
     3           XNM1,UMNELC,UELC,TCL,EA,ER,EC,E1,E2,E3,E4,E5,E6
      COMMON / SYMBOL /  DCT,BLANK,XXX
      DOUBLE PRECISION  DX,DX2,DXSQR,REYNLD,DEY,EGACEL,XLAMDA,TIME,DT,DTT,PI,XN,
     1           BSQR,REYNLD,DEY,EGACEL,XLAMDA,TIME,DT,DTT,PI,XN,
     2           EKMEAN,EKTUB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,E2,
     3           XNM1,UMNELC,UELC,TCL,EA,ER,EC,E1,E2,E3,E4,E5,E6
      DIMENSION  PHI(64,201),GAMA(64,201),T(64,201),TURBKE(64,201),
      DOUBLE PRECISION  UMEAN(201),TEMP(201),A(128),S(32),XALPHA(64),
     1           CHILAM(201),CAMEAN(201),X(64),Y(201),E(32),
     2           BETA(201),GA(201),TMU(64),TKV(32),IAD(64),IAP(64)
      DIMENSION  IPC(64),TMC(64),TAB(201),MM(3)
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C     THIS ROUTINE COMPUTES THE KINETIC ENERGY OF THE DISTURBANCE
C     MOTION AS A FUNCTION OF Y AND WAVE NUMBER IN THE X DIRECTION.
C     WAVE OR MODE NUMBER ONE CORresponds TO THE MEAN VALUE OF THE
C     DISTURBANCE. THE SPECTRUM OF THE TURBULENT ENERGY IS CONTAINED
C     IN MODE NUMBERS TWO THROUGH N+1. THE SPECTRAL VALUES
C     ARRAY EG IS USED FOR STORAGE OF THE SPECTRAL VALUES
C
      IF(ICOUNT .EQ. KSTEP) RETURN
      IFIG=1
      IF(INDEX .LT. 0) GO TO 40
      L=M+1
      DO 10 J=2,NM1
      JM=J-1
      JP=J+1
      EC(1,J)=EC*(PHI(1,JM)-PHI(1,JP))**2
      EG(L,J)=EC*(PHI(M,JM)-PHI(M,JP))**2
      DO 10 I=2,MH
      LL=IAD(I)
5
```

```
      LP=LL+1                                                                   0001 0430
      LM=LL-1                                                                   0001 0440
                                                                                0001 0450
      FG(I,J)=FA*(FB*((PHI(LM,JM)**2+PHI(LL,JM)**2+PHI(LM,JP)**2                0001 0460
     1                +PHI(LL,JP)**2)-2.D0*(PHI(LM,JP)*PHI(LM,JM)+              0001 0470
     2                 PHI(LL,JP)*PHI(LL,JM)))                                  0001 0480
     3              +FE(I)*(PHI(LM,J)**2+PHI(LL,J)**2))                         0001 0490
   10 CONTINUE                                                                  0001 0500
C                                                                               0001 0510
CC    MEAN VALUES                                                               0001 0520
C                                                                               0001 0530
      T(1)=0.D0                                                                 0001 0540
      DO 20 I=1,MHP                                                             0001 0550
      DO 15 J=2,NM1                                                             0001 0560
      T(J)=FG(I,J)                                                              0001 0570
   15 CONTINUE                                                                  0001 0580
      CALL ASUM(T,NM1,SUM)                                                      0001 0590
      ETA(I)=SUM/XNM1                                                           0001 0600
      GA(I)=FG(I,NH)                                                            0001 0610
   20 CONTINUE                                                                  0001 0620
C                                                                               0001 0630
CC    TITLE                                                                     0001 0640
C                                                                               0001 0650
      L=ICOUNT+1                                                                0001 0660
      WRITE(6,22) L                                                             0001 0670
   22 FORMAT(10X,'ENERGY SPECTRUM FOR CYCLE NUMBER ',I5)                        0001 0680
      WRITE(6,21)                                                               0001 0690
   21 FORMAT(14X,'MEAN SPECTRUM',36X,'MODE',20X,'MIDSTREAM SPECTRUM')           0001 0700
      CALL PLOTSP(ETA,GA,MHP)                                                   0001 0710
      DO 30 I=1,MHP                                                             0001 0720
      ETA(I)=FG(I,16)                                                           0001 0730
      GA(I)=FG(I,186)                                                           0001 0740
   30 CONTINUE                                                                  0001 0750
      WRITE(6,23) Y(16),Y(186)                                                  0001 0760
   23 FORMAT(10X,'SPECTRUM FOR Y=',F6.4,32X,'MODE',18X,'SPECTRUM FOR Y='       0001 0770
     1       ,F6.4)                                                           0001 0780
      CALL PLOTSP(ETA,GA,MHP)                                                   0001 0790
C                                                                               0001 0800
C      PRINT THE ENERGY IN SEVERAL MODES AS FUNCTIONS OF Y.                     0001 0810
      ETA(1)=0.D0                                                               0001 0820
      ETA(N)=0.D0                                                               0001 0830
      GA(1)=0.D0                                                                0001 0840
      GA(N)=0.D0                                                                0001 0850
      I1=2                                                                      0001 0860
      I2=3                                                                      0001 0870
      DO 39 KL=1,2                                                              0001 0880
      DO 35 J=2,NM1                                                             0001 0890
      ETA(J)=FG(I1,J)                                                           0001 0900
```

```
      GA(J)=EG(I2,J)                                                    0001C910
   35 CONTINUE                                                          0001C920
      WRITE(6,36) I1,I2                                                 0001C930
   36 FORMAT(14X,'ENERGY IN MODE ',I2,35X,'J',20X,'ENERGY IN MODE ',I2) 0001C940
      CALL PLOTSP(ETA,GA,N)                                             0001C950
      I1=4                                                              0001C960
      I2=5                                                              0001C970
   39 CONTINUE                                                          0001C980
      GO TO 100                                                         0001C990
C                                                                       0001D000
CC    SIMILAR OPERATIONS FOR THE SPECTRUM OF SQUARED VORTICITY          0001D010
C                                                                       0001D020
   40 E6=1.DC/XM**2                                                     0001D030
      DO 50 J=1,N                                                       0001D040
      EG(1,J)=E6*PHI(1,J)**2                                            0001D050
      EG(MHP,J)=E6*PHI(M,J)**2                                          0001D060
      DO 50 I=2,MH                                                      0001D070
      LL=IAO(I)                                                         0001D080
      EG(I,J)=E6*(PHI(LL-1,J)**2+PHI(LL,J)**2)                          0001D090
   50 CONTINUE                                                          0001D100
C                                                                       0001D110
      DO 60 I=1,MHP                                                     0001D120
      DO 55 J=1,N                                                       0001D130
      T(J)=EG(I,J)                                                      0001D140
   55 CONTINUE                                                          0001D150
      CALL ASUM(T,N,SUM)                                                0001D160
      ETA(I)=SUM/XNM1                                                   0001D170
      GA(I)=EG(I,NH)                                                    0001D180
   60 CONTINUE                                                          0001D190
C                                                                       0001D200
CC    TITLE                                                             0001D210
C                                                                       0001D220
      L=ICOUNT+1                                                        0001D230
      WRITE(6,61) L                                                     0001D240
   61 FORMAT(10X,'VORTICITY SPECTRUM FOR CYCLE NUMBER  ',I5)            0001D250
      WRITE(6,21)                                                       0001D260
      CALL PLOTSP(ETA,GA,MHP)                                           0001D270
      DO 70 I=1,MHP                                                     0001D280
      ETA(I)=EG(I,16)                                                   0001D290
      GA(I)=EG(I,186)                                                   0001D300
   70 CONTINUE                                                          0001D310
      WRITE(6,23) Y(16),Y(186)                                          0001D320
      CALL PLOTSP(ETA,GA,MHP)                                           0001D330
  100 RETURN                                                            0001D340
      END                                                               0001D350
```

```
      SUBROUTINE PLOTSP(AX,BX,NN)                                       00011360
      COMMON / SYMBOL / DOT,BLANK,XXX                                   00011370
      COMMON / PICTUR / ALINE(49),BLINE(49),APLOT(201),BPLOT(201)       00011380
      DOUBLE PRECISION  AX(1),BX(1)                                     00011390
C                                                                       00011400
C     THIS ROUTINE PRINTS ONLINE PLOTS OF THE POSITIVE NN-DIMENSIONAL   00011410
C     VECTORS AX AND BX.                                                00011420
C                                                                       00011430
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00011440
C     PUT AX & BX IN APLOT & BPLOT                                      00011450
C                                                                       00011460
      DO 10 I=1,NN                                                      00011470
      APLOT(I)=AX(I)                                                    00011480
      BPLOT(I)=BX(I)                                                    00011490
   10 CONTINUE                                                          00011500
C                                                                       00011510
C     FIND LARGEST ELEMENT IN APLOT & BPLOT                             00011520
C                                                                       00011530
      ABIG=0.                                                           00011540
      BBIG=0.                                                           00011550
      DO 20 I=1,NN                                                      00011560
      ABIG=AMAX1(ABIG,ABS(APLOT(I)))                                    00011570
      BBIG=AMAX1(BBIG,ABS(BPLOT(I)))                                    00011580
   20 CONTINUE                                                          00011590
C                                                                       00011600
C     SET UP TOP HORIZONTAL AXIS                                        00011610
C                                                                       00011620
      DO 30 I=1,49                                                      00011630
      BLINE(I)=DOT                                                      00011640
   30 CONTINUE                                                          00011650
      WRITE(6,31) ALINE,BLINE                                           00011660
      ALINE(I)=DOT                                                      00011670
   31 FORMAT(14X,49A1,20X,49A1)                                         00011680
C                                                                       00011690
C     BLANK ALINE & BLINE. PUT PLOTTING SYMBOL IN COMPUTED ELEMENT      00011700
C                                                                       00011710
      DO 40 I=2,49                                                      00011720
      ALINE(I)=BLANK                                                    00011730
      BLINE(I)=BLANK                                                    00011740
   40 CONTINUE                                                          00011750
      ALINE(1)=DOT                                                      00011760
      BLINE(1)=DOT                                                      00011770
C                                                                       00011780
      DO 50 I=1,NN                                                      00011790
      JA=APLOT(I)/ABIG*48.+1.                                           00011800
      JB=BPLOT(I)/BBIG*48.+1.                                           00011810
```

```
      ALINE(JA)=XXX                                                     00011920
      BLINE(JB)=XXX                                                     00011930
      WRITE(6,51) APLOT(I),ALINE,I,BPLOT(I),BLINE                       00011940
   51 FORMAT(1X,F10.4,2X,49A1,2X,I3,2X,F10.4,2X,49A1)                   00011950
      ALINE(JA)=BLANK                                                   00011960
      BLINE(JB)=BLANK                                                   00011970
      ALINE(1)=DOT                                                      00011980
      BLINE(1)=DOT                                                      00011990
   50 CONTINUE                                                          00012000
C                                                                       00012010
CC    SET UP BOTTOM HORIZONTAL AXIS                                     00012020
C                                                                       00012030
      DO 60 I=1,49                                                      00012040
      ALINE(I)=DOT                                                      00012050
      BLINE(I)=DOT                                                      00012060
   60 CONTINUE                                                          00012070
      WRITE(6,61) ALINE,BLINE                                           00012080
   61 FORMAT(14X,49A1,20X,49A1//)                                       00012090
      RETURN                                                            00012000
      END                                                               00012010
```

```
      SUBROUTINE PRINT                                                  00001 20200
      COMMON / FIELDS /  PHI,GAMA,EG,GAMOLD                             00001 20400
      COMMON / VECTOR /  T,X,XAL,DHA,DHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00001 20600
     1 COMMON / ARRAYS /  U,PHILAM,ETA,CA,X,Y                           00001 20800
      COMMON / PARAMS /  IPC,IMC,INV,IAC,NM1,NM2,NM3,NV,NH,MP1,MP2,MP3,MM1,  00001 21000
     1  N,M,NP1,NP2,NPP,MH,MMP1,ICOUNT,ITEL,INT,KSTEP,ISTART,          00001 21200
     2  MSCIP,ITAPE,NITER,NITMAX,IFIG,MCOMP,                           00001 21400
     3  KINT                                                           00001 21600
C                                                                       00001 21800
      COMMON / FACTOR /  DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR,  00001 22000
     1  DSQRD,REY,EPA,CELL,XLAMDA,TIME,DT,DTT,PI,H,                    00001 22200
     2  EKMEAN,EXTUB,UVRAR,ALBHA,BETA,XM,XN                            00001 22400
C                                                                       00001 22600
      COMMON / SYMBOL /  XNM1,UMNELC,TOL,EPS,EC,E1,E2,E3,E4,E5,E6     00001 22800
     1  ,XXX                                                           00001 23000
C                                                                       00001 23200
      DIMENSION    PHI(24,201),UMEAN(201),TEMP(24,201),TUBKE(24,201),  00001 23400
     1 DX(24),DX2(24),DXSQR(24),DXSQR4(24),DXSQR8(24),DY(24),DY2(24),   00001 23600
     2 DYSQR(24),R(24),RSQR(24),DSQRD(24),REY(24),EPA(24),CELL(24),     00001 23800
     3 XLAMDA(201),TIME(201),DT(201),DTT(201),PI(24),H(24),             00001 24000
     4 EKMEAN(201),EKTUB(24),UVRAR(24),ALBHA(24),BETA(24),XM(24),       00001 24200
     5 XN(24),UMNELC(24),XXX(24),TOL(24),EPS(24),EC(24),E1(24),E2(24),  00001 24400
     6 E3(24),E4(24),E5(24),E6(24),ETA(201),CA(201),X(64,201),          00001 24600
     7 Y(32,201),INV(32),IAC(64),IAP(64)                                00001 24800
C                                                                       00001 25000
      DIMENSION    IPC(24),TMC(24),NV(24),IAC(24),NM1(24),NM2(24),     00001 25200
     1 NM3(24),NP1(24),NP2(24),NPP(24),MH(24),MMP1(24),MP1(24),         00001 25400
     2 MP2(24),MP3(24),MM1(24),NH(24)                                   00001 25600
C                                                                       00001 25800
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00001 26000
C                                                                       00001 26200
C     THIS ROUTINE PRINTS THE VALUES OF ICOUNT, T, DT, EKMEAN, EKTUB, EKTOT, UVRAR, UMNELC,  00001 26400
C     NITER AND NITOT                                                   00001 26600
C     CALL ENERGY                                                       00001 26800
C                                                                       00001 27000
      IF(IFIG.GT.0)WRITE(6,11)                                          00001 27200
      EKTOT=EKMEAN+EKTURB                                               00001 27400
      WRITE(6,12) ICOUNT,TIME,DT,EKMEAN,EKTURB,UELC,EKTOT,UVRAR,UMNELC, 00001 27600
     1  NITER,NITOT                                                     00001 27800
   11 FORMAT(3X,'CYCLE',4X,'TIME',9X,'DT',7X,'MEAN KE',5X,'TURB KE',7X, 00001 28000
     1 'UVRAR',7X,'UMNELC',3X,                                          00001 28200
     2 'NITOT',2X,'NITEL')                                              00001 28400
   12 FORMAT(3X,I5,3X,E10.6,2(3X,E13.7),3X,E9.7,3X,E10.4,3X,            00001 28600
     1 I8,6X,2X,I5,2X,I4)                                             00001 28800
      IFIG=-1                                                           00001 29000
      RETURN                                                            00001 29200
      END                                                               00001 29400
```

```
      SUBROUTINE MEANS                                                  0001 2450
      COMMON / FIELDS /  PHI,GAMA,EG,GAMOLD                             0001 2460
      COMMON / VECTOR /  T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,  0001 2470
     1                   U,PHILAM,EE,ETA,GA,X,Y                         0001 2480
      COMMON / ARRAYS /  IPO,IMO,INV,IAO,IAP,MM(3)                      0001 2490
      COMMON / PARAMS /  N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 0001 2500
     1                   M2M,MH,MHP,NPPOR,ICOUNT,ITEL,INT,YSTEP,ISTART, 0001 2510
     2                   ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IEIG,MCOMP, 0001 2520
     3                   KINT                                           0001 2530
      COMMON / FACTOR /  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 0001 2540
     1                   RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H, 0001 2550
     2                   EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN 0001 2560
     3                  ,XNM1,UMNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6  0001 2570
      COMMON / SYMBOL /  DOT,BLANK,XXX                                  0001 2580
      DOUBLE PRECISION   DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 0001 2590
     1                   RSQRP,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H, 0001 2600
     2                   EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN 0001 2610
     3                  ,XNM1,UMNELO,UELO,TOL,EA,EB,EC,E1,E2,E3,E4,E5,E6  0001 2620
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  0001 2630
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        0001 2640
     1                   PHILAM(201),T(201),A(129),S(32),XALPHA(64),     0001 2650
     2                   PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),    0001 2660
     3                   ETA(201),GA(201)                               0001 2670
      DIMENSION          IPO(64),IMO(64),INV(32),IAO(64),IAP(64)        0001 2680
                                                                        0001 2690
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 0001 2700
C                                                                       0001 2710
C     THIS ROUTINE COMPUTES THE MEAN VALUES OF PHI AND GAMA             0001 2720
C     ARRAY A IS USED FOR TEMPORARY STORAGE.                            0001 2730
C                                                                       0001 2740
      DO 10 J=1,N                                                       0001 2750
      DO 20 I=1,M                                                       0001 2760
      A(I)=PHI(I,J)                                                     0001 2770
   20 CONTINUE                                                          0001 2780
      CALL ASUM(A,M,SUM)                                                0001 2790
      PHMEAN(J)=SUM/XM                                                  0001 2800
      DO 30 I=1,M                                                       0001 2810
      A(I)=GAMA(I,J)                                                    0001 2820
   30 CONTINUE                                                          0001 2830
      CALL ASUM(A,M,SUM)                                                0001 2840
      GAMEAN(J)=SUM/XM+3.D0*Y(J)                                        0001 2850
   10 CONTINUE                                                          0001 2860
      RETURN                                                            0001 2870
      END                                                               0001 2880
```

```
      SUBROUTINE EDDYS(INDEX)                                           00001200
C     COMMON  / FIELDS  / PHI,GAMA,EG,GAMOLD                           00001210
C     COMMON  / VECTOR  / T,A,S,XALPHA,GAMEAN,GAMEAN,UMEAN,TEMP,TURBKE, 00001220
C                         U,P,U1,U2,EE,EE1,EE2,X,Y                    00001230
C     COMMON  / ARRAYS  / TP,TMP,NV,TAB,MM(3)                           00001240
C     COMMON  / PARAMS  / N,M,NP2,ND2,NM2,NMX,NH,MP1,MP2,MP3,MM1,      00001250
C                         M2M,MH,MHD,NP1HB,COUNT,ITEL,INT,KEIEP,ISTART, 00001260
C                         ISCID,MSCID,ITAB,NTER,NITT,NITMAX,IFIS,MCOMP, 00001270
C                         KTMP                                          00001280
      COMMON  / FACTOR  / DX2,DXSQP,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00001290
     1                    RSQR8,REY,REYNL,CF,XLAMDA,TMEAN,CFT,PI,H,     00001300
     2                    REKME,N,UKNELO,VELO,EA,EB,EC,E1,E2,E3,E4,E5,E6 00001310
      COMMON  / SYMBOL  / DX,DX2,REYNLO,XX,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00001320
     1                    RSQR8,REY,REYNLO,CF,XLAMDA,TMEAN,CFT,PI,H,    00001330
     2                    REKMEAN,REKTUB,UVP,ITL,EA,EB,EC,E1,E2,E3,E4,E5,E6, 00001340
     3                    EXMIT,UKNELO,XX                               00001350
      DIMENSION   PHI(64,201),GAMA(64,201),T(201),A(201),S(201),        00001360
     1            DUDY(201),DVDY(201),DUDX(201),DVDX(201),              00001370
     2            PHME(201),C(201),X(64),Y(201),EE(32),                 00001380
     3            EA(201),EB(201),EC(201)                               00001390
      DIMENSION   IPC(64),IMC(64),TNV(32),IAC(64),IAB(64)               00001400
C                                                                       00001410
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00001420
C                                                                       00001430
C     ASSUMES SUBROUTINE MEANS HAS GENERATED CURRENT VALUES OF UMEAN    00001440
C     AND GAMEAN                                                        00001450
C                                                                       00001460
C     IF INDEX > 0 ROUTINE PUTS TURBULENT COMPONENTS OF STREAM          00001470
C     FUNCTION IN ARRAY EG                                              00001480
C                                                                       00001490
C     IF INDEX < 0 ROUTINE PUTS TURBULENT COMPONENTS OF VORTICITY       00001500
C     IN ARRAY EG                                                       00001510
C                                                                       00001520
      IF(INDEX .LT. 0) GO TO 15                                         00001530
    1 DO 100 J=1,N                                                      00001540
      DO 100 I=1,M                                                      00001550
  100 EG(I,J)=PHI(I,J)-UMEAN(J)                                         00001560
   10 CONTINUE                                                          00001570
      GO TO 25                                                          00001580
   15 DO 200 J=1,N                                                      00001590
      DO 200 I=1,M
```

```
      FG(I,J)=GAMA(I,J)-GAMEAN(J)                                       00013350
   20 CONTINUE                                                          00013360
   25 RETURN                                                            00013370
      END                                                               00013380
```

```
      SUBROUTINE ENERGY                                                 00001300
      COMMON / FIELDS/  PHI,GAMA,EG,GAMOLD                              00001400
      COMMON / VECTOR /  T,AS,XAL,EE,ALPHA,DUMEAN,GAMEAN,UMEAN,TEMP,TURBKE,  00001500
     1                   U,DPHI,AM,EE,TA,CA,X,Y                         00001600
      COMMON / ARRAYS /  ID,C,TIME,TINV,FAD,MV(3)                       00001700
      COMMON / PARAMS /  N,M,NO1,NO2,NM1,NV2,NM3,NW4,NH,ND1,MP2,MP3,MM1,  00001800
     1                   M2M,MH,MHD,NCRB,IC,COUNT,TTEL,TNT,KSTEP,IST,ART  00001900
     2                   ISCID,MSCID,EE,NITER,NITCT,NITMAX,FEIG,MCOMP,  00002000
     3                   KENT                                           00002100
      COMMON / FACTOR /  DX2,DXSQRD,DXSQRD4,DXSQRB,DY,DY2,DYSQR,R,RSQR,  00002200
     1                   RSQRD,R,PSI,XMU,EL,EA,EB,EC,E1,E2,EB,EA,EE,E6  00002300
      COMMON / SYMBOL /  XNU1,UVN,EL,TCL,EA,EB,EC,E1,E2,EB,EA,EE,E6  00002400
     1                   XCC,RLAM,XXX,DYSQRD,DYSQRB,DY2,DYSQR,R,RSQR,  00002500
     2                   EKMEAN,XNU,EL,TCL,EA,EB,EC,E1,E2,EB,EA,EE,E6  00002600
      COMMON / PRECISION /  EK2,DXSQRD,DXSQRB,DY,DY2,DYSQR,R,RSQR,  00002700
     1                   RSQRD,R,PSI,XMU,EL,EA,EB,EC,E1,E2,EB,EA,EE,E6  00002800
     2                   EKMEAN,XNU1,UVN,EL,TCL,EA,EB,EC,E1,E2,EB,EA,EE,E6  00002900
      DOUBLE PRECISION  PHI(64,20),GAMA(20),GAMOLD(20),GAMEAN(20),UMEAN(20),  00003000
     1                  PHIAM(20),CA(20),TEMP(20),TURBKE(20)           00003100
      DOUBLE PRECISION  XM2                                             00003200
      DIMENSION  PHI(64,20),GAMA(20),GAMOLD(20),GAMEAN(20),UMEAN(20),  00003300
     1           PHIAM(20),CA(20),TEMP(20),TURBKE(20)                 00003400
      DOUBLE PRECISION  XM2                                             00003500
C                                                                       00003600
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00003700
C                                                                       00003800
C     THIS ROUTINE COMPUTES THE AVERAGE KINETIC ENERGIES OF BOTH THE    00003900
C     MEAN AND THE TURBULENT COMPONENTS OF THE FLOW                     00004000
C                                                                       00004100
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00004200
C                                                                       00004300
      CALL EDDYS(10)                                                    00004400
      CALL STRESS(T)                                                    00004500
C     MEAN FLOW, T IS USED FOR TEMPORARY STORAGE                        00004600
      DO 10 J=2,NV1                                                     00004700
      T(J)=UMEAN(J)**2                                                  00004800
   10 CONTINUE                                                          00004900
      T(1)=0.                                                           00005000
      CALL ASUM(T,NM1,SUM)                                              00005100
      EKMEAN=SUM/(2.*DO*XNM2)                                           00005200
C                                                                       00005300
C     TURBULENT FLOW, TURBULENT COMPONENTS OF PHI ARE STORED IN EG BY   00005400
C     EDDYS, ARRAY A IS USED FOR TEMPORARY STORAGE.                     00005500
```

```
      XM2=2.D0*XM                                                                    0001 3850
      TURBKE(N)=0.                                                                   0001 3860
      TURBKE(1)=0.                                                                   0001 3870
      DO 30 J=2,NM1                                                                  0001 3880
      DO 20 I=1,M                                                                    0001 3890
      IP=IPO(I)                                                                      0001 3900
      IM=IMO(I)                                                                      0001 3910
      A(I)=((EG(I,J+1)-EG(I,J-1))/DY2)**2+((EG(IP,J)-EG(IM,J))/DX2)**2               0001 3920
   20 CONTINUE                                                                       0001 3930
      CALL ASUM(A,M,SUM)                                                             0001 3940
      TURBKE(J)=SUM/XM2                                                              0001 3950
   30 CONTINUE                                                                       0001 3960
      CALL ASUM(TURBKE,NM1,SUM)                                                      0001 3970
      EKTURB=SUM/XNM1                                                                0001 3980
      RETURN                                                                         0001 3990
      END                                                                            0001 4000
```

```
      SUBROUTINE STRESS                                                 00001400
      COMMON / FIELDS /  PHI,GAMA,EG,CANOLD                             00001410
      COMMON / VECTOR /  T,A,S,XALPHA,DUMEAN,GAMEAN,UMEAN,TEMP,TURBKE,  00001420
     1                   U,PHILAM,EE,FA,G,X,Y                           00001430
      COMMON / ARRAYS /  ID,TMC,NV,TAC,MM(3)                            00001440
      COMMON / PARAMS /  N,M,ND1,ND2,NM1,NM2,NV2,NV4,NH,MP1,MP2,MP3,MM1,00001450
     1                   M2M,MH,MUD,NCOMB,ICOUNT,ITEL,INT,KSTEP,IST,ADT,00001460
     2                   ISCIP,MSCIP,IADE,NTIER,NITER,NITMAX,IEIG,MCOMP,00001470
     3                   KFNT                                           00001480
      COMMON / FACTOR /  DX,DX2,DXSQR,DXSQR4,DY,DY2,DYSQR,R,RSQR,      00001490
     1                   RSQRD,BEYNLD,BEYSQR,ACEL,XLAMDA,TIME,DT,DTI,H, 00001500
     2                   EKMEAN,EKTUB,UVBAR,ATER,ALPHA,BETA,XM,EG,      00001510
     3                   XNMI,UMELD,TEL,EA,ER,EC,E1,E2,E3,E4,E5,E6      00001520
      COMMON / SYMBOL /  DDX,DDX2,DDXSQR,DDXSQR4,DDXSQRD,DDY,DDY2,DDYSQR,00001530
      DOUBLE PRECISION   DDXSQR,BEYSQR,EKMEAN,BEYNLD,UVBAR,ATER,XNMI,  00001540
     1                   UMELD,PHI(64,201),GAMA(64,201),TMP(201),TUBKY, 00001550
     2                   UMEAN(201),T(201),A(201),S(201),XALPHA(201),   00001560
     3                   PHILAM(201),TEMP(201),GAMEAN(201),X(64),Y(201),00001570
     4                   ETA(201),TMC(64),TNV(32),TAC(64),TAB(64)       00001580
      DIMENSION          TD(64),MM(3)                                   00001590
C                                                                       00001600
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * 00001610
C                                                                       00001620
C     ASSUMES THAT FLUCTUATING COMP. OF PHI IS STORED IN EC             00001630
C     COMPUTES REYNOLDS STRESSES AND MEAN VELOCITIES.                   00001640
C                                                                       00001650
      TEMP(1)=0.                                                        00001660
      TEMP(N)=0.                                                        00001670
      DO 20 J=2,NM1                                                     00001680
      DO 10 I=1,M                                                       00001690
      IP=I+1                                                            00001700
      TEMP(I)=0.                                                        00001710
      A(I)=(I-1.)*DY*EG(I,J-1)-EG(I,J+1))*(1.-DY*EG(IP,J)-EG(IM,J))     00001720
   10 CONTINUE                                                          00001730
      CALL ASUM(A,M,SUM)                                                00001740
      TEMP(J)=SUM/BTERM                                                 00001750
   20 CONTINUE                                                          00001760
      CALL ASUM(TEMP,NM1,SUM)                                           00001770
      UVBAR=SUM/XNM1                                                    00001780
C                                                                       00001790
C     MEAN VELOCITY                                                     00001800
C                                                                       00001810
```

```
      DO 30 J=2,NM1                                                     00014470
      UMEAN(J)=(PHMEAN(J-1)-PHMEAN(J+1))/DY2+U(J)                       00014480
   30 CONTINUE                                                          00014490
      UMEAN(1)=0.                                                       00014500
      UMEAN(N)=0.                                                       00014510
      CALL ASUM(UMEAN,NM1,SUM)                                          00014520
      UMNELO=SUM/XNM1                                                   00014530
      UELO=(2.D0+PHMEAN(2)-PHMEAN(NM1)+PHILAM(2)-PHILAM(NM1))/4.D0      00014540
      UELO=DLOG(EKTURB)                                                 00014550
      RETURN                                                            00014560
      END                                                               00014570

      SUBROUTINE ASUM(A,NN,SUM)                                         00014580
      DOUBLE PRECISION A(1)                                             00014590
      DOUBLE PRECISION SUMM                                             00014600
C                                                                       00014610
C     THIS ROUTINE COMPUTES THE SUM OF THE FIRST NN ELEMENTS OF A,      00014620
C     AND STORES THE RESULT IN SUM                                      00014630
C                                                                       00014640
      SUMM=0.D0                                                         00014650
      DO 10 I=1,NN                                                      00014660
      SUMM=SUMM+A(I)                                                    00014670
   10 CONTINUE                                                          00014680
      SUM=SUMM                                                          00014690
      RETURN                                                            00014700
      END                                                               00014710
```

```
      SUBROUTINE PLOT                                                   00014720
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              00014730
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   00014740
     1                  U,PHILAM,EE,ETA,GA,X,Y                         00014750
      COMMON / ARRAYS / IPO,IMO,INV,IAO,IAP,MM(3)                       00014760
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1, 00014770
     1                  M2M,MH,MHP,NPEPS,ICOUNT,ITEL,INT,KSTEP,ISTART, 00014780
     2                  ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP, 00014790
     3                  KINT                                           00014800
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00014810
     1                  RSQR8,REYNLD,DFY,EPACEL,XLAMDA,TIME,DT,DTT,PI,H, 00014820
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN 00014830
     3                 ,XNM1,UMNELD,UFLD,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6 00014840
      COMMON / SYMBOL / DOT,BLANK,XXX                                   00014850
      COMMON / PICTUR / ALINE(49),BLINE(49),APLOT(201),BPLOT(201)       00014860
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR, 00014870
     1                  RSQR8,REYNLD,DFY,EPACEL,XLAMDA,TIME,DT,DTT,PI,H, 00014880
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN 00014890
     3                 ,XNM1,UMNELD,UFLD,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6 00014900
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  00014910
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        00014920
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     00014930
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),    00014940
     3                  ETA(201),GA(201)                                00014950
      DIMENSION         IPO(64),IMO(64),INV(32),IAO(64),IAP(64)         00014960
C                                                                       00014970
C     THIS ROUTINE PRINTS ONLINE PLOTS OF MEAN VELOCITY,VORTICITY,      00014980
C     TURBULENT KINETIC ENERGY AND REYNOLDS STRESS AS FUNCTIONS OF Y.   00014990
C                                                                       00015000
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   00015010
C                                                                       00015020
      IFIG=1                                                            00015030
      III=-1                                                            00015040
C                                                                       00015050
C     PUT UMEAN IN APLOT , TEMP IN BPLOT                                00015060
C                                                                       00015070
      DO 10 J=1,N                                                       00015080
      APLOT(J)=UMEAN(J)                                                 00015090
      BPLOT(J)=TEMP(J)                                                  00015100
   10 CONTINUE                                                          00015110
C                                                                       00015120
C     TITLE                                                             00015130
C                                                                       00015140
      WRITE(6,11)                                                       00015150
   11 FORMAT(1HO,4X,'UMEAN',25X,'UMEAN',34X,'UVBAR',28X,'UVBAR')        00015160
```

```
C                                                                       0001 5180
C      FIND LARGEST ELEMENTS IN APLOT AND BPLOT                         0001 5190
C                                                                       0001 5200
   15 ABIG=0.                                                           0001 5210
      BBIG=0.                                                           0001 5220
      DO 20 J=1,N                                                       0001 5230
      ABIG=AMAX1(ABIG,ABS(APLOT(J)))                                    0001 5240
      BBIG=AMAX1(BBIG,ABS(BPLOT(J)))                                    0001 5250
   20 CONTINUE                                                          0001 5260
C                                                                       0001 5270
C      SET UP TOP HORIZONTAL AXIS                                       0001 5280
C                                                                       0001 5290
      DO 30 I=1,49                                                      0001 5300
      ALINE(I)=DOT                                                      0001 5310
      BLINE(I)=DOT                                                      0001 5320
   30 CONTINUE                                                          0001 5330
      WRITE(6,12)ALINE,BLINE                                            0001 5340
   12 FORMAT(1HX,49A1,20X,49A1)                                         0001 5350
C                                                                       0001 5360
C      BLANK ALINE AND BLINE AND PUT PLOTTING SYMBOL IN COMPUTED ELEMENT 0001 5370
C                                                                       0001 5380
      DO 40 I=1,49                                                      0001 5390
      ALINE(I)=BLANK                                                    0001 5400
      BLINE(I)=BLANK                                                    0001 5410
   40 CONTINUE                                                          0001 5420
      ALINE(1)=DOT                                                      0001 5430
      BLINE(25)=DOT                                                     0001 5440
      J=N                                                               0001 5450
      DO 50 JC=1,N,2                                                    0001 5460
      JA=APLOT(J)/ABIG*48.+1.                                           0001 5470
      JB=BPLOT(J)/BBIG*24.+25.                                          0001 5480
      ALINE(JA)=XXX                                                     0001 5490
      BLINE(JB)=XXX                                                     0001 5500
      WRITE(6,13)APLOT(J),ALINE,J,BPLOT(J),BLINE                        0001 5510
   13 FORMAT(1X,E10.4,2X,49A1,2X,I3,3X,E10.4,2X,49A1)                   0001 5520
      ALINE(JA)=BLANK                                                   0001 5530
      BLINE(JB)=BLANK                                                   0001 5540
      ALINE(1)=DOT                                                      0001 5550
      BLINE(25)=DOT                                                     0001 5560
      J=J-2                                                             0001 5570
   50 CONTINUE                                                          0001 5580
C                                                                       0001 5590
C      SET UP BOTTOM HORIZONTAL AXIS                                    0001 5600
C                                                                       0001 5610
      DO 60 I=1,49                                                      0001 5620
      ALINE(I)=DOT                                                      0001 5630
      BLINE(I)=DOT                                                      0001 5640
   60 CONTINUE                                                          0001 5650
```

```
      WRITE(6,14)ALINE,BLINE                                                    0001566C
   14 FORMAT(14X,49A1,20X,49A1//)                                               0001567C
      IF(III .GT. 0)GO TO 100                                                   0001568C
C                                                                               0001569C
      PUT TURBKE IN APLOT , GAMEAN IN BPLOT                                     0001570C
C                                                                               0001571C
   55 DO 70 J=1,N                                                               0001572C
      APLOT(J)=TURBKE(J)                                                        0001573C
      BPLOT(J)=GAMEAN(J)                                                        0001574C
   70 CONTINUE                                                                  0001575C
      III=1                                                                     0001576C
C                                                                               0001577C
      TITLE                                                                     0001578C
C                                                                               0001579C
      WRITE(6,16)                                                               0001580C
   16 FORMAT(1H0,4X,'TURBKE',25X,'TURBKE',32X,'GAMEAN',26X,'GAMEAN')            0001581C
      GO TO 15                                                                  0001582C
  100 RETURN                                                                    0001583C
      END                                                                       0001584C
```

```
      SUBROUTINE CPLOT(A,M,N)                                                   00016830
      DIMENSION LINE(115),ISYMB(42),BV(21),A(N,M)                               00016840
C     SUBROUTINE CPLOT(A,M,N,T)                                                 00016850
CC    DIMENSION T(24)                                                           00016860
CC  THIS SUBROUTINE PLOTS A(I,J) , DIMENSION M,N, WITH TITLE T                  00016870
C                                                                               00016880
      DATA ISYMB/1H ,1HO,1H ,1HA,1H ,1HB,1H ,1HC,1H ,1HD,1H ,1HE,               00016890
     1,1H ,1HF,1H ,1HG,1H ,1HH,1H ,1HI,1H ,1HJ,1H ,1HQ,1H ,1HR,1H ,1HS,         00016900
     2,1H ,1HT,1H ,1HU,1H ,1HV,1H ,1HW,1H ,1HX,1H ,1HY,1H ,1HZ/                 00016910
      DATA IBLK/1H /                                                            00016920
      AMIN=1.E+70                                                               00016930
      DO 6 I=1,N                                                                00016940
      DO 5 J=1,M                                                                00016950
    5 AMIN=AMIN1(AMIN,A(I,J))                                                   00016960
    6 CONTINUE                                                                  00016970
      DO 8 I=1,N                                                                00016980
      DO 7 J=1,M                                                                00016990
    7 A(I,J)=A(I,J)-AMIN                                                        00017000
    8 CONTINUE                                                                  00017010
      AMAX=-1.E+70                                                              00017020
      DO 11 I=1,N                                                               00017030
      DO 10 J=1,M                                                               00017040
   10 AMAX=AMAX1(AMAX,A(I,J))                                                   00017050
   11 CONTINUE                                                                  00017060
      RGE=AMAX                                                                  00017070
      IF(ABS(RGE)-1.0E-10) 12,12,13                                             00017080
   12 WRITE(6,1000)                                                             00017090
      WRITE(6,1400) RGE,AMAX,AMIN                                               00017100
 1400 FORMAT(//,5X,13H THE RANGE IS,E15.6,/,5X,13H   MAXIMUM IS,E15.6,/,        00017110
     15X,13H   MINIMUM IS,F15.6)                                                00017120
      RETURN                                                                    00017130
   13 CONTINUE                                                                  00017140
      SCF=40./AMAX                                                              00017150
      DO 21 I=1,N                                                               00017160
      DO 20 J=1,M                                                               00017170
   20 A(I,J)=SCF*A(I,J)                                                         00017180
   21 CONTINUE                                                                  00017190
      DO 30 I=1,21                                                              00017200
      AI=FLOAT(I-1)                                                             00017210
   30 BV(I)=(2.*AI/SCF)+AMIN                                                    00017220
      WRITE(6,1000)                                                             00017230
C     WRITE(6,113)                                                              00017240
C 113 FORMAT('0', T10, 2A4)                                                     00017250
      WRITE(6,1050)                                                             00017260
      WRITE(6,1300) BV(1),BV(1),BV(21)                                          00017270
      WRITE(6,1100) BV(2),BV(3),BV(4),BV(5),BV(6)                               00017280
```

```
      WRITE(6,1200) BV(7),BV(8),BV(9),BV(10),BV(11)                     00017290
      WRITE(6,1210) BV(12),BV(13),BV(14),BV(15),BV(16)                  00017300
      WRITE(6,1220) BV(17),BV(18),BV(19),BV(20),BV(21)                  00017310
      WRITE(6,1050)                                                     00017320
      M1=M+1                                                            00017330
      DO 50 K=1,N                                                       00017340
      DO 40 J=1,M                                                       00017350
      AJ=A(K,J)+2.50001                                                 00017360
      JJ=INT(AJ)                                                        00017370
   40 LINE(J)=ISYMB(JJ)                                                 00017380
      IF(M1-115) 41,41,46                                               00017390
   41 CONTINUE                                                          00017400
      DO 45 I=M1,115                                                    00017410
   45 LINE(I)=IBLK                                                      00017420
   46 CONTINUE                                                          00017430
      WRITE(6,2000) LINE                                                00017440
   50 CONTINUE                                                          00017450
      DO 61 I=1,N                                                       00017460
      DO 60 J=1,M                                                       00017470
   60 A(I,J)=A(I,J)/SCF +AMIN                                           00017480
   61 CONTINUE                                                          00017490
 1000 FORMAT(1H1)                                                       00017500
 1050 FORMAT(//)                                                        00017510
 1100 FORMAT(5X,3H A=,F8.3,3X,3H B=,F8.3,3X,2H C=,F8.3,3X,3H D=,F8.3,   00017520
     13X,2H E=,F8.3)                                                    00017530
 1200 FORMAT(5X,3H F=,F8.3,3X,2H G=,F8.3,3X,2H H=,F8.3,3X,3H I=,F8.3,   00017540
     13X,2H J=,F8.3)                                                    00017550
 1210 FORMAT(5X,3H O=,F8.3,3X,2H R=,F8.3,3X,3H S=,F8.3,3X,3H T=,F8.3,   00017560
     13X,2H U=,F8.3)                                                    00017570
 1220 FORMAT(5X,3H V=,F8.3,3X,3H W=,F8.3,3X,3H X=,F8.3,3X,2H Y=,F8.3,   00017580
     13X,3H Z=,F8.3)                                                    00017590
 1300 FORMAT(5X,3H  O=,F10.6,5X,7H RANGE(,E15.6,5X,E15.6,2H ))          00017600
 2000 FORMAT(2X,115A1)                                                  00017610
      END                                                               00017620
```

```
      SUBROUTINE OUTPUT(INDEX)                                                  0001585C
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                                      0001586C
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,           0001587C
     1                  U,PHILAM,EE,ETA,GA,X,Y                                 0001588C
      COMMON / ARRAYS / IPO,IMO,INV,IAO,IAP,MM(3)                               0001589C
      COMMON / PARAMS / N,M,NP1,NP2,NM1,NM2,NM3,NM4,NH,MP1,MP2,MP3,MM1,         0001590C
     1                  M2M,MH,MH2,MPOPP,ICOUNT,ITEL,INT,KSTEP,ISTART,         0001591C
     2                  ISCIP,MSCIP,ITAPE,NITER,NITOT,NITMAX,IFIG,MCOMP,       0001592C
     3                  KINT                                                   0001593C
      COMMON / FACTOR / DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR,         0001594C
     1                  RSQR2,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,       0001595C
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN       0001596C
     3                 ,XNM1,UMNELD,UELD,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6        0001597C
      COMMON / SYMBOL / DOT,BLANK,XXX                                           0001598C
      DOUBLE PRECISION  DX,DX2,DXSQR,DXSQR4,DXSQR8,DY,DY2,DYSQR,R,RSQR,         0001599C
     1                  RSQR2,REYNLD,REY,FRACEL,XLAMDA,TIME,DT,DTT,PI,H,       0001600C
     2                  EKMEAN,EKTURB,UVBAR,ATERM,BTERM,ALPHA,BETA,XM,XN       0001601C
     3                 ,XNM1,UMNELD,UELD,TOL,FA,FB,FC,F1,F2,F3,F4,F5,F6        0001602C
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)          0001603C
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),              0001604C
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),            0001605C
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),           0001606C
     3                  ETA(201),GA(201)                                       0001607C
      DIMENSION         IPO(64),IMO(64),INV(32),IAO(64),IAP(64)                 0001608C
C     * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *   0001609C
C                                                                               0001610C
C     IF INDEX=1 , OUTPUT IS ARRAY PHI                                          0001611C
C     IF INDEX=2 , OUTPUT IS ARRAY GAMA                                         0001612C
C     IF INDEX=3 , OUTPUT IS ARRAY EG                                           0001613C
C                                                                               0001614C
      IF(INDEX .GT. 1)GO TO 21                                                  0001615C
C                                                                               0001616C
C     PRINT ARRAY PHI                                                           0001617C
C                                                                               0001618C
    1 WRITE(6,11)                                                               0001619C
      NLST=M                                                                    0001620C
      NFRST=1                                                                   0001621C
   10 ILST=MINO(NLST,10)                                                        0001622C
      KLST=NFRST+ILST-1                                                         0001623C
      WRITE(6,14)(I,I=NFRST,KLST)                                               0001624C
      WRITE(6,15)(X(I),I=NFRST,KLST)                                            0001625C
      DO 20 JC=1,N                                                              0001626C
      J=N-JC+1                                                                  0001627C
      WRITE(6,16)J,Y(J),(PHI(I,J),I=NFRST,KLST)                                 0001628C
   20 CONTINUE                                                                  0001629C
```

```
      WRITE(6,17)
      NLST=NLST-10
      NERST=NERST+10
      IF(NLST .GT. 0)GO TO 10
      GO TO 100
   21 IF(INDEX .GT. 2)GO TO 41
C
C     PRINT ARRAY GAMA
C
   22 WRITE(6,12)
      NLST=M
      NERST=1
   30 ILST=MINO(NLST,10)
      KLST=NERST+ILST-1
      WRITE(6,14)(I,I=NERST,KLST)
      WRITE(6,15)(X(I),I=NERST,KLST)
      DO 40 JC=1,N
      J=N-JC+1
      WRITE(6,16)J,Y(J),(GAMA(I,J),I=NERST,KLST)
   40 CONTINUE
      WRITE(6,17)
      NLST=NLST-10
      NERST=NERST+10
      IF(NLST .GT. 0)GO TO 30
      GO TO 100
C
C     PRINT ARRAY EG
C
   41 WRITE(6,13)
      NLST=M
      NERST=1
   50 ILST=MINO(NLST,10)
      KLST=NERST+ILST-1
      WRITE(6,14)(I,I=NERST,KLST)
      WRITE(6,15)(X(I),I=NERST,KLST)
      DO 60 JC=1,N
      J=N-JC+1
      WRITE(6,16)J,Y(J),(EG(I,J),I=NERST,KLST)
   60 CONTINUE
      WRITE(6,17)
      NERST=NERST+10
      NLST=NLST-10
      IF(NLST .GT. 0)GO TO 50
   11 FORMAT(1H ,5X,'** PHI **')
   12 FORMAT(1H ,5X,'** GAMA **')
   13 FORMAT(1H ,5X,'** EG **')
   14 FORMAT(1H ,5X,'I=',12X,10(I3,8X))
   15 FORMAT(1H ,5X,'X=',11X,10(F6.3,5X))
```

```
0001 6310
0001 6320
0001 6330
0001 6340
0001 6350
0001 6360
0001 6370
0001 6380
0001 6390
0001 6400
0001 6410
0001 6420
0001 6430
0001 6440
0001 6450
0001 6460
0001 6470
0001 6480
0001 6490
0001 6500
0001 6510
0001 6520
0001 6530
0001 6540
0001 6550
0001 6560
0001 6570
0001 6580
0001 6590
0001 6600
0001 6610
0001 6620
0001 6630
0001 6640
0001 6650
0001 6660
0001 6670
0001 6680
0001 6690
0001 6700
0001 6710
0001 6720
0001 6730
0001 6740
0001 6750
0001 6760
0001 6770
0001 6780
```

```
 16 FORMAT(1X,'J=',I3,1X,'Y=',F6.3,2X,10(F10.7,1X))                    00016760
 17 FORMAT(1H0,'***  ***  ***  ***  ***  ***  ***  ***  ***  ***'//)   00016900
100 RETURN                                                            00016970
    END                                                               00016920
```

```
      SUBROUTINE LOAD                                                   0001 7630
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              0001 7640
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   0001 7650
     1                  U,PHILAM,EE,ETA,GA,X,Y                         0001 7660
      COMMON / ARRAYS / IPO,IMO,INV,IAC,IAP,MM(2)                       0001 7670
      COMMON / PARAMS / NUMBER(31)                                      0001 7680
      COMMON / FACTOR / FACTO                                           0001 7690
      COMMON / SYMBOL / DOT,BLANK,XXX                                   0001 7700
      DOUBLE PRECISION  FACTO(42)                                       0001 7710
      DIMENSION     PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)  0001 7720
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        0001 7730
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     0001 7740
     2                  PHMEAN(201),GAMEAN(201),X(64),Y(201),EE(32),    0001 7750
     3                  ETA(201),GA(201)                                0001 7760
      DIMENSION         IPO(64),IMO(64),INV(32),IAC(64),IAP(64)         0001 7770
      ITAPE=NUMBER(25)                                                  0001 7780
      REWIND ITAPE                                                      0001 7790
      READ(ITAPE)  PHI,GAMA,EG,GAMOLD                                   0001 7800
      READ(ITAPE)  T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP                0001 7810
      READ(ITAPE)  TURBKE,U,PHILAM,EE,ETA,GA,X,Y                        0001 7820
      READ(ITAPE)  IPO,IMO,INV,IAC,IAP,MM                               0001 7830
      READ(ITAPE)  NUMBER                                               0001 7840
      READ(ITAPE)  FACTO                                                0001 7850
      READ(ITAPE) DOT,BLANK,XXX                                         0001 7860
      REWIND ITAPE                                                      0001 7870
      CALL DHARM(A,MM,INV,S,0,IERR)                                     0001 7880
      RETURN                                                            0001 7890
      END                                                               0001 7900
```

```
      SUBROUTINE RELOAD                                                 0001 7810
      COMMON / FIELDS / PHI,GAMA,EG,GAMOLD                              0001 7820
      COMMON / VECTOR / T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP,TURBKE,   0001 7830
     1                  U,PHILAM,EE,ETA,GA,X,Y                         0001 7840
      COMMON / ARRAYS / IPO,IMO,INV,IAO,IAP,MM(3)                       0001 7850
      COMMON / PARAMS / NUMBER(31)                                      0001 7860
      COMMON / FACTOR / FACTO                                           0001 7870
      COMMON / SYMBOL / DOT,BLANK,XXX                                   0001 7880
      DOUBLE PRECISION  FACTO(42)                                       0001 7890
      DIMENSION    PHI(64,201),GAMA(64,201),EG(64,201),GAMOLD(64,201)   0001 8000
      DOUBLE PRECISION  UMEAN(201),TEMP(201),TURBKE(201),U(201),        0001 8010
     1                  PHILAM(201),T(201),A(128),S(32),XALPHA(64),     0001 8020
     2                  PHMEAN(201),GAMEAN(201),X(44),Y(201),EE(32),    0001 8030
     3                  ETA(201),GA(201)                                0001 8040
      DIMENSION         IPO(64),IMO(64),INV(32),IAO(64),IAP(64)         0001 8050
      ITAPE=NUMBER(25)                                                  0001 8060
      REWIND ITAPE                                                      0001 8070
      WRITE(ITAPE) PHI,GAMA,EG,GAMOLD                                   0001 8080
      WRITE(ITAPE) T,A,S,XALPHA,PHMEAN,GAMEAN,UMEAN,TEMP                0001 8090
      WRITE(ITAPE) TURBKE,U,PHILAM,EE,ETA,GA,X,Y                        0001 8100
      WRITE(ITAPE) IPO,IMO,INV,IAO,IAP,MM                               0001 8110
      WRITE(ITAPE) NUMBER                                               0001 8120
      WRITE(ITAPE) FACTO                                                0001 8130
      WRITE(ITAPE) DOT,BLANK,XXX                                        0001 8140
      REWIND ITAPE                                                      0001 8150
      RETURN                                                            0001 8160
      END                                                               0001 8170
```

APPENDIX C

LISTING OF THE FORTRAN CODE FOR THE SOLUTION OF THE ALGEBRAIC EIGENVALUE PROBLEM

```
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / B,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      COMMON / EXTRA / NUMVEC
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      DOUBLE PRECISION  XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      XLAMDA=.200
      FRACEL=.400
      NUMVEC=4
      N=201
      M=64
      REYNLD=25000.D0
      ALPHA=1.D0
      CALL SETUP
      CALL EXACT
   11 FORMAT(2F20.14)
      STOP
      END


      SUBROUTINE SETUP
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / B,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      COMMON / EXTRA / NUMVEC
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      DOUBLE PRECISION  XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
C
```

```
C       MISCELANEOUS PARAMETERS
C
        WRITE(6,99)
        H=3.14159265358980
        NM1=N-1
        NM2=N-2
        NM3=N-3
        NM4=N-4
        NM5=N-5
        NH=(N+1)/2
        NHM1=NH-1
        NHM2=NH-2
        NHM3=NH-3
        NHM4=NH-4
        NHM5=NH-5
        XN=DFLOAT(N)
        XNM1=DFLOAT(NM1)
        XNH=DFLOAT(NH)
        XNHM1=DFLOAT(NHM1)
        DX=2.0D0*H/DFLOAT(M)
        DY=2.0D0/XNM1
        DXSQR=DX*DX
        DYSQR=DY*DY
        RATIO=DYSQR/DXSQR
        RATIOP=RATIO+1.0D0
        DTT=FRACEL*DSQRT(DX*DY)/1.5D0
        DT=XLAMDA*DX*DY*REYNLD/4.0D0
        DT=DMIN1(DT,DTT)
        ITEL=1
        IF(DT .EQ. DTT) ITEL=-1
C
CCC     VORTICITY BOUNDARY CONDITIONS
C
        P1=8.0D0
        P2=-3.0D0
        P3=8.0D0/9.0D0
        P4=-.125D0
        P1=4.0D0
        P2=-.5D0
        P3=0.0D0
        P4=0.0D0
C
CCC     TITLE
C
        WRITE(6,13) N,M,ITEL
        WRITE(6,14) DX,DXSQR,RATIO
        WRITE(6,15) DY,DYSQR,RATIOP
        WRITE(6,16) DT,FRACEL,XLAMDA
```

```
      WRITE(6,17) H,ALPHA,REYNLD
   13 FORMAT(10X,'N= ',I3,8X,'M=      ',I3,8X,'ITEL=   ',I3)
   14 FORMAT(10X,'DX=',F8.6,3X,'DXSQR= ',F8.6,3X,'RATIO= ',F8.5)
   15 FORMAT(10X,'DY=',F8.6,3X,'DYSQR= ',F8.6,3X,'RATIOP=',F8.5)
   16 FORMAT(10X,'DT=',F8.6,3X,'FRACEL=',F8.6,3X,'XLAMDA=',F8.6)
   17 FORMAT(10X,'H= ',F8.6,3X,'ALPHA= ',F8.6,3X,'REYNLD=',F10.4)
      DO 10 J=1,N
      Y(J)=DY*DFLOAT(J-1)-1.D0
      U(J)=-1.5D0*(Y(J)**2-1.D0)
   10 CONTINUE
      WRITE(6,99)
   99 FORMAT(1HO,10X,'* * * * * * * * * * * * * * * * * * * * * * * * *
     1* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *//)
      RETURN
      END


      SUBROUTINE EXACT
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / B,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      COMMON / EXTRA / NUMVEC
      DOUBLE PRECISION  XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
C
      OMEGA=-2.D0-(ALPHA*DY)**2
      ZETA=ALPHA
      OMEGT=OMEGA+2.D0*RATIOP
      CALL SETMAT
      DETRM=-2.D0*RATIOP/(REYNLD*DYSQR)
      DO 10 I=1,NCAL
      VAL(1,I)=VAL(1,I)+DETRM
      TEMP(1,I)=TEMP(1,I)+DETRM
      VAL(1,I)=VAL(1,I)/ALPHA
      VAL(2,I)=VAL(2,I)/ALPHA
   10 CONTINUE
      GAMA=TEMP(1,1)
      BETA=TEMP(2,1)
```

```
      C1=-TEMP(1,1)/ALPHA
      C2=-TEMP(2,1)/ALPHA
      D1=1.D0
      D2=0.D0
      WRITE(6,1)
    1 FORMAT(10X,'*** CONTINUOUS CASE ***     DATA FOR LEAST STABLE MODE
     1'//)
      WRITE(6,2) OMEGA,OMEGT,ZETA
      WRITE(6,3) D1,D2
      WRITE(6,4) C1,C2
      WRITE(6,5) GAMA,BETA,ALPHA
    2 FORMAT(10X,'OMEGA= ',F13.8,3X,'OMEGT= ',F13.8,3X,'ZETA= ',F13.8)
    3 FORMAT(10X,'D1    = ',F13.8,3X,'D2    = ',F13.8)
    4 FORMAT(10X,'C1    = ',F13.8,3X,'C2    = ',F13.8)
    5 FORMAT(10X,'GAMA  = ',F13.8,3X,'BETA  = ',F13.8,3X,'ALPHA=',F13.8//)
      IF(NUMVEC .GE. 0) CALL OUTPUT(NUMVEC)
      RETURN
      END


      SUBROUTINE DISCRT
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / B,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      COMMON / EXTRA / NUMVEC
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      DOUBLE PRECISION  XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      DOUBLE PRECISION XX,YY
      OMEGA=-2.D0-4.D0*RATIO*DSIN(ALPHA*DX/2.D0)**2
      OMEGT=OMEGA+2.D0*RATIOP
      ZETA=DSIN(ALPHA*DX)/DX
      CALL SETMAT
      FAC=2.D0*RATIOP/(REYNLD*DYSQR)
      IF(NCAL .EQ. 0) GO TO 100
    8 DO 20 J=1,NCAL
      C1=TEMP(1,J)
      C2=TEMP(2,J)
      PUNT=C1-FAC
```

```
      CRIT=C2
      DO 10 I=1,100
      XX=CRIT*DT
      YY=PUNT*DT
      D1=YY/(DSINH(YY)*DCOS(XX))*(C1-FAC*DCOSH(YY)*DCOS(XX))
      D2=XX/(DSIN(XX)*DCOSH(YY))*(C2-FAC*DSINH(YY)*DSIN(XX))
      PUNT=1.DO-DABS(PUNT/D1)
      CRIT=1.DO-DABS(CRIT/D2)
      IF((PUNT .LT. .00001) .AND. (CRIT .LT. .00001)) GO TO 15
    9 PUNT=D1
      CRIT=D2
   10 CONTINUE
      I=100
   15 TEMP(1,J)=D1
      TEMP(2,J)=D2
      IF(I .GT. 99) WRITE(6,16) J,I
   16 FORMAT(10X,'J=',I3,3X,I3,'ITERATIONS')
   20 CONTINUE
      GAMA=TEMP(1,1)
      BETA=TEMP(2,1)
      C1=-GAMA/ALPHA
      C2=-BETA/ALPHA
      D1=DCOSH(GAMA*DT)*DCOS(BETA*DT)
      D2=DSINH(GAMA*DT)*DSIN(BETA*DT)
      WRITE(6,1)
    1 FORMAT(10X,'*** DISCRETE CASE ***     DATA FOR LEAST STABLE MODE')
      WRITE(6,2) OMEGA,OMEGT,ZETA
      WRITE(6,3) D1,D2
      WRITE(6,4) C1,C2
      WRITE(6,5) GAMA,BETA,ALPHA
    2 FORMAT(10X,'OMEGA= ',F13.8,3X,'OMEGT= ',F13.8,3X,'ZETA= ',F13.8)
    3 FORMAT(10X,'D1   = ',F13.8,3X,'D2   = ',F13.8)
    4 FORMAT(10X,'C1   = ',F13.8,3X,'C2   = ',F13.8)
    5 FORMAT(10X,'GAMA = ',F13.8,3X,'BETA = ',F13.8,3X,'ALPHA=',F13.8//)
      IF(NUMVEC .GE. 0) CALL OUTPUT(NUMVEC)
  100 RETURN
      END


      SUBROUTINE SETMAT
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / P,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
```

```
     3                    FAC,PUNT,CRIT,P1,P2,P3,P4
      COMMON / EXTRA / NUMVEC
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      DOUBLE PRECISION  XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
C
C     ZERO ARRAYS
C
      DO 10 I=1,NHM1
      VAL(1,I)=0.D0
      VAL(2,I)=0.D0
      TEMP(1,I)=0.D0
      TEMP(2,I)=0.D0
      DO 10 J=1,NHM1
      W(1,I,J)=0.D0
      W(2,I,J)=0.D0
      B(I,J)=0.D0
   10 CONTINUE
C
C     SET UP AND INVERT MATRIX B
C
      DO 20 I=1,NHM2
      B(I,I)=OMEGA
      B(I,I+1)=1.D0
      B(I+1,I)=1.D0
   20 CONTINUE
      B(NHM1,NHM1)=OMEGA
      B(NHM1,NHM2)=2.D0
      WRITE(6,22)
   22 FORMAT(1H0,10X,'ENTER MINV')
      CALL MINV(B,NHM1,DETRM,IORD,INUM)
      WRITE(6,23)
   23 FORMAT(10X,'EXIT MINV'//)
      IF(DETRM .LT. .10-06) WRITE(6,21) DETRM
   21 FORMAT(1H0,10X,'B ALMOST SINGULAR. DETRM= ',E10.4//)
C
C     SET UP MATRIX W
C
      DETRM=-3.D0*ZETA*DYSQR
      DO 30 I=1,NHM2
      W(2,I,I)=ZETA*U(I+1)*OMEGA-DETRM
      W(2,I+1,I)=ZETA*U(I+2)
      W(2,I,I+1)=ZETA*U(I+1)
   30 CONTINUE
      W(2,NHM1,NHM2)=2.D0*ZETA*U(NH)
```

```
      W(2,NHM1,NHM1)=ZETA*U(NH)*OMEGA
      DO 40 I=1,NHM1
      DO 40 J=1,NHM1
      DO 40 K=1,NHM1
      W(1,I,J)=W(1,I,J)+B(I,K)*W(2,K,J)
   40 CONTINUE
      DO 50 I=1,NHM1
      DO 50 J=1,NHM1
      W(2,I,J)=W(1,I,J)
      W(1,I,J)=0.D0
   50 CONTINUE
      DETRM=1.D0/(REYNLD*DYSQR)
      DO 60 I=1,NHM2
      W(1,I,I)=DETRM*OMEGT
      W(1,I+1,I)=DETRM
      W(1,I,I+1)=DETRM
   60 CONTINUE
      W(1,NHM1,NHM1)=DETRM*OMEGT
      W(1,NHM1,NHM2)=2.D0*DETRM
      DO 70 I=1,NHM1
      W(1,I,1)=W(1,I,1)+B(I,1)*P1*DETRM
      W(1,I,2)=W(1,I,2)+B(I,1)*P2*DETRM
      W(1,I,3)=W(1,I,3)+B(I,1)*P3*DETRM
      W(1,I,4)=W(1,I,4)+B(I,1)*P4*DETRM
   70 CONTINUE
C
CC    NORMALIZE MATRIX W
C
      FAC=DSQRT(W(1,NHM1,NHM1)**2+W(2,NHM1,NHM1)**2)
      FAC=1.D0/FAC
      DO 75 I=1,NHM1
      DO 75 J=1,NHM1
      W(1,I,J)=W(1,I,J)*FAC
      W(2,I,J)=W(2,I,J)*FAC
   75 CONTINUE
C
CC     OBTAIN THE EIGENVALUES AND EIGENVECTORS OF W
C
      WRITE(6,72)
   72 FORMAT(10X,'ENTER DALMAT')
      CALL DALMAT(W,VAL,NHM1,NHM1,NCAL)
      WRITE(6,71) NCAL
   71 FORMAT(1HO,10X,'***  ',I3,' EIGENVALUES AND EIGENVECTORS CALCULATE
     1D'//)
C
CC    REMOVE THE NORMALIZING FACTOR FROM THE COMPUTED EIGENVALUES
C
      DO 80 I=1,NCAL
```

```
      VAL(1,I)=VAL(1,I)/FAC
      VAL(2,I)=VAL(2,I)/FAC
      VAL(2,I)=-VAL(2,I)
   80 CONTINUE
C
CCCCCC FIND THE STABILITY ORDER OF THE EIGENVALUES --- MAX REAL PART
CCCCCC ARRAY IORD(I)=STORAGE LOCATION OF EIGENVALUE OR EIGENVECTOR
CCCCCC                WITH STABILITY NUMBER=I
CCCCCC ARRAY INUM(I)=STABILITY NUMBER OF EIGENVALUE OR EIGENVECTOR
CCCCCC                WITH STORAGE NUMBER=I
C
      DO 100 IC=1,NCAL
      PUNT=-.1D35
      DO 90 I=1,NCAL
      IF(VAL(1,I) .GT. PUNT) II=I
      PUNT=DMAX1(PUNT,VAL(1,I))
   90 CONTINUE
      IORD(IC)=II
      INUM(II)=IC
C
CCC   PUT EIGENVALUES IN ARRAY TEMP IN STABILITY ORDER
C
      TEMP(1,IC)=VAL(1,II)
      TEMP(2,IC)=VAL(2,II)
      VAL(1,II)=-.1D35
  100 CONTINUE
      DO 110 I=1,NCAL
      II=INUM(I)
      VAL(1,I)=TEMP(1,II)
  110 CONTINUE
C
CCC   NORMALIZE THE EIGENVECTORS
C
      IF(NUMVEC .LE. 0) GO TO 130
  115 DO 126 K=1,NUMVEC
      JJ=IORD(K)
      PUNT=-W(2,NHM1,JJ)/W(1,NHM1,JJ)
      DO 120 I=1,NHM1
      VECT(1,I)=W(1,I,JJ)-PUNT*W(2,I,JJ)
      VECT(2,I)=W(2,I,JJ)+PUNT*W(1,I,JJ)
  120 CONTINUE
      PUNT=0.D0
      PUNT=VECT(1,NHM1)
      PUNT=1.D0/PUNT
      DO 126 I=1,NHM1
      W(1,I,JJ)=PUNT*VECT(1,I)
      W(2,I,JJ)=PUNT*VECT(2,I)
  126 CONTINUE
```

```
  130 RETURN
      END


      SUBROUTINE OUTPUT(NUMVEC)
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / B,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      DOUBLE PRECISION  XN,XNM1,XN4,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACEL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
C
CCC   PRINT THE EIGENVALUES
C
      WRITE(6,11)
   11 FORMAT(5X,'STORNUM  STABNUM',10X,'EIGENVALUE',23X,'STORNUM  STABNU
     1M',10X,'EIGENVALUE')
      DO 10 I=1,NCAL
      WRITE(6,12) I,INUM(I),VAL(1,I),VAL(2,I),IORD(I),I,TEMP(1,I),
     1            TEMP(2,I)
   12 FORMAT(7X,I3,6X,I3,3X,E13.7,' + ',E13.7,' *I',12X,I3,6X,I3,3X,
     1       E13.7,' + ',E13.7,' *I')
   10 CONTINUE
C
CCC   PRINT EIGENVECTORS
C
      IF(NUMVEC .GT. NCAL) NUMVEC=NCAL
      IF(NUMVEC .LT. 1) GO TO 30
   15 NSTART=1
      NN=4
   19 NNN=MIN0(NUMVEC,NN)
      WRITE(6,13) (I,I=NSTART,NNN)
   13 FORMAT(11X,3('STABNUM = ',I3,17X),'STABNUM = ',I3)
      WRITE(6,14)
   14 FORMAT(4(9X,'REAL',8X,'IMAGINARY'))
      DO 20 I=1,NHM1
      WRITE(6,16) (W(1,I,IORD(J)),W(2,I,IORD(J)),J=NSTART,NNN)
   20 CONTINUE
      WRITE(6,99)
```

```
      NN=NN+4
      NSTART=NSTART+4
      IF(NSTART .LT. NUMVEC) GO TO 19
   16 FORMAT(1X,4(4X,E12.6,2X,E12.6))
   99 FORMAT(1HO,10X,'* * * * * * * * * * * * * * * * * * * * * * * * * *
     1* * * * * * * * * * * * * * * * * * * * * * * * * '//)
   30 RETURN
      END


      SUBROUTINE PRINT(INDEX)
      COMMON / FIELDS / W,VAL,Y,U
      COMMON / SPACES / B,BI,TEMP,VECT
      COMMON / ARRAYS / IORD(101),INUM(101)
      COMMON / PARAMS / M,N,NM1,NM2,NM3,NM4,NM5,NH,NHM1,NHM2,NHM3,NHM4,
     1                  NHM5,KOUNT,MODE,NCAL
      COMMON / FACTOR / XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACFL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      COMMON / EXTRA / NUMVEC
      DOUBLE PRECISION  W(2,100,100),VAL(2,100),Y(201),U(201)
      DOUBLE PRECISION  B(100,100),BI(100,100),TEMP(2,100),VECT(2,100)
      DOUBLE PRECISION  XN,XNM1,XNH,XNHM1,DX,DY,DXSQR,DYSQR,RATIO,
     1                  RATIOP,DT,DTT,FRACFL,XLAMDA,ALPHA,H,REYNLD,
     2                  ZETA,OMEGA,OMEGT,C1,C2,D1,D2,GAMA,BETA,DETRM,
     3                  FAC,PUNT,CRIT,P1,P2,P3,P4
      NSTART=1
      IF(INDEX .EQ. 3) GO TO 10
    5 IF(INDEX .EQ. 1) K=1
      IF(INDEX .EQ. 2) K=2
      GO TO 30
   10 WRITE(6,1)
    1 FORMAT(1HO,10X,'*** MATRIX B ***'//)
   15 NSTOP=NSTART+9
      WRITE(6,11) (J,J=NSTART,NSTOP)
   11 FORMAT(1HO,' J=',8X,I3,9(9X,I3))
      DO 20 I=1,NHM1
      WRITE(6,12) I,(B(I,J),J=NSTART,NSTOP)
   12 FORMAT(1X,'I=',I3,10(2X,E10.4))
   20 CONTINUE
      NSTART=NSTART+10
      IF(NSTART .GT. NHM1) GO TO 100
   25 GO TO 15
   30 WRITE(6,2) K
    2 FORMAT(1HO,10X,'*** MATRIX W *** K=',I3,' ***'//)
   35 NSTOP=NSTART+9
      WRITE(6,11) (J,J=NSTART,NSTOP)
```

```
      DO 40 I=1,NHM1
      WRITE(6,12) I,(N(K,I,J),J=NSTART,NSTOP)
   40 CONTINUE
      NSTART=NSTART+10
      IF(NSTART .GT. NHM1) GO TO 100
   45 GO TO 35
  100 RETURN
      END


      SUBROUTINE MINV(A,N,D,L,M)                                          MINV 330
      DIMENSION A(1),L(1),M(1)                                            MINV 340
      DOUBLE PRECISION A,D,BIGA,HOLD                                      MINV 420
C                                                                         MINV  10
C                                                                         MINV  30
C        PURPOSE                                                          MINV  60
C           INVERT A MATRIX                                               MINV  70
C                                                                         MINV  80
C        USAGE                                                            MINV  90
C           CALL MINV(A,N,D,L,M)                                          MINV 100
C                                                                         MINV 110
C        DESCRIPTION OF PARAMETERS                                        MINV 120
C           A - INPUT MATRIX, DESTROYED IN COMPUTATION AND REPLACED BY    MINV 130
C                 RESULTANT INVERSE.                                      MINV 140
C           N - ORDER OF MATRIX A                                         MINV 150
C           D - RESULTANT DETERMINANT                                     MINV 160
C           L - WORK VECTOR OF LENGTH N                                   MINV 170
C           M - WORK VECTOR OF LENGTH N                                   MINV 180
C                                                                         MINV 190
C                                                                         MINV 250
C        METHOD                                                           MINV 260
C           THE STANDARD GAUSS-JORDAN METHOD IS USED. THE DETERMINANT     MINV 270
C           IS ALSO CALCULATED. A DETERMINANT OF ZERO INDICATES THAT      MINV 280
C           THE MATRIX IS SINGULAR.                                       MINV 290
C                                                                         MINV 300
C ......................................................................  MINV 310
C                                                                         MINV 530
C        SEARCH FOR LARGEST ELEMENT                                       MINV 540
C                                                                         MINV 550
      D=1.0                                                               MINV 560
      NK=-N                                                               MINV 570
      DO 80 K=1,N                                                         MINV 580
      NK=NK+N                                                             MINV 590
      L(K)=K                                                              MINV 600
      M(K)=K                                                              MINV 610
      KK=NK+K                                                             MINV 620
      BIGA=A(KK)                                                          MINV 630
      DO 20 J=K,N                                                         MINV 640
```

```
      IZ=N*(J-1)                                                        MINV  650
      DO 20 I=K,N                                                       MINV  660
      IJ=IZ+I                                                           MINV  670
   10 IF(DABS(BIGA)-DABS(A(IJ))) 15,20,20                               MINV  680
   15 BIGA=A(IJ)                                                        MINV  690
      L(K)=I                                                            MINV  700
      M(K)=J                                                            MINV  710
   20 CONTINUE                                                          MINV  720
C                                                                       MINV  730
C        INTERCHANGE ROWS                                               MINV  740
C                                                                       MINV  750
      J=L(K)                                                            MINV  760
      IF(J-K) 35,35,25                                                  MINV  770
   25 KI=K-N                                                            MINV  780
      DO 30 I=1,N                                                       MINV  790
      KI=KI+N                                                           MINV  800
      HOLD=-A(KI)                                                       MINV  810
      JI=KI-K+J                                                         MINV  820
      A(KI)=A(JI)                                                       MINV  830
   30 A(JI) =HOLD                                                       MINV  840
C                                                                       MINV  850
C        INTERCHANGE COLUMNS                                            MINV  860
C                                                                       MINV  870
   35 I=M(K)                                                            MINV  880
      IF(I-K) 45,45,38                                                  MINV  890
   38 JP=N*(I-1)                                                        MINV  900
      DO 40 J=1,N                                                       MINV  910
      JK=NK+J                                                           MINV  920
      JI=JP+J                                                           MINV  930
      HOLD=-A(JK)                                                       MINV  940
      A(JK)=A(JI)                                                       MINV  950
   40 A(JI) =HOLD                                                       MINV  960
C                                                                       MINV  970
C        DIVIDE COLUMN BY MINUS PIVOT (VALUE OF PIVOT ELEMENT IS        MINV  980
C        CONTAINED IN BIGA)                                             MINV  990
C                                                                       MINV1000
   45 IF(BIGA) 48,46,48                                                 MINV1010
   46 D=0.0                                                             MINV1020
      RETURN                                                            MINV1030
   48 DO 55 I=1,N                                                       MINV1040
      IF(I-K) 50,55,50                                                  MINV1050
   50 IK=NK+I                                                           MINV1060
      A(IK)=A(IK)/(-BIGA)                                               MINV1070
   55 CONTINUE                                                          MINV1080
C                                                                       MINV1090
C        REDUCE MATRIX                                                  MINV1100
C                                                                       MINV1110
      DO 65 I=1,N                                                       MINV1120
```

```
      IK=NK+I                                                          MINV1130
      HOLD=A(IK)                                                       MINV1140
      IJ=I-N                                                           MINV1150
      DO 65 J=1,N                                                      MINV1160
      IJ=IJ+N                                                          MINV1170
      IF(I-K) 60,65,60                                                 MINV1180
   60 IF(J-K) 62,65,62                                                 MINV1190
   62 KJ=IJ-I+K                                                        MINV1200
      A(IJ)=HOLD*A(KJ)+A(IJ)                                           MINV1210
   65 CONTINUE                                                         MINV1220
C                                                                      MINV1230
CC         DIVIDE ROW BY PIVOT                                         MINV1240
C                                                                      MINV1250
      KJ=K-N                                                           MINV1260
      DO 75 J=1,N                                                      MINV1270
      KJ=KJ+N                                                          MINV1280
      IF(J-K) 70,75,70                                                 MINV1290
   70 A(KJ)=A(KJ)/BIGA                                                 MINV1300
   75 CONTINUE                                                         MINV1310
C                                                                      MINV1320
CC         PRODUCT OF PIVOTS                                           MINV1330
C                                                                      MINV1340
      D=D*BIGA                                                         MINV1350
C                                                                      MINV1360
CC         REPLACE PIVOT BY RECIPROCAL                                 MINV1370
C                                                                      MINV1380
      A(KK)=1.0/BIGA                                                   MINV1390
   80 CONTINUE                                                         MINV1400
C                                                                      MINV1410
CC         FINAL ROW AND COLUMN INTERCHANGE                            MINV1420
C                                                                      MINV1430
      K=N                                                              MINV1440
  100 K=(K-1)                                                          MINV1450
      IF(K) 150,150,105                                                MINV1460
  105 I=L(K)                                                           MINV1470
      IF(I-K) 120,120,108                                              MINV1480
  108 JQ=N*(K-1)                                                       MINV1490
      JR=N*(I-1)                                                       MINV1500
      DO 110 J=1,N                                                     MINV1510
      JK=JQ+J                                                          MINV1520
      HOLD=A(JK)                                                       MINV1530
      JI=JR+J                                                          MINV1540
      A(JK)=-A(JI)                                                     MINV1550
  110 A(JI) =HOLD                                                      MINV1560
  120 J=M(K)                                                           MINV1570
      IF(J-K) 100,100,125                                              MINV1580
  125 KI=K-N                                                           MINV1590
      DO 130 I=1,N                                                     MINV1600
```

```
      KI=KI+N                                                          MINV1610
      HOLD=A(KI)                                                       MINV1620
      JI=KI-K+J                                                        MINV1630
      A(KI)=-A(JI)                                                     MINV1640
  130 A(JI) =HOLD                                                      MINV1650
      GO TO 100                                                        MINV1660
  150 RETURN                                                           MINV1670
      END                                                              MINV1680


      SUBROUTINE DALMAT(A,LAMBDA,M,IA,NCAL)                            DALMA044
      IMPLICIT REAL*8 (A-H), REAL*8 (O-Z)                              DALMA052
      COMMON / SPACES / HL,MULT,VECT
      COMMON / EXTRA / NUMVEC
      COMPLEX*16 CSQRQ, CONJQ                                          DALMA053
      REAL*8 REAQ,AIMAQ,CABQ,SQRQ,FLOAQ,ABQ                            DALMA054
      COMPLEX*16 HL(100,100),H(100,100),MULT(100),VECT(100)
      COMPLEX*16 A(IA,1),LAMBDA(1),SHIFT(3),TEMP,SIN,COS,TEMP1,TEMP2
      LOGICAL INTH(100),TWICE
      INTEGER INT(100),R,RP1,RP2
C                                                                      DALMA045
C                                                                      DALMA047
C  PROG. AUTHORS JOHN RINZEL,R.E.FUNDERLIC,UNION CARBIDE CORP.         DALMA048
C  NUCLEAR DIVISION,CENTRAL DATA PROCESSING FACILITY,                  DALMA049
C  OAK RIDGE TENNESSEE                                                 DALMA050
C                                                                      DALMA051
C    PURPOSE                                                           DALMA006
C        DALMAT CALCULATES IN DOUBLE-PRECISION THE EIGENVALUES AND EIGENDALMA007
C        VECTORS OF AN ARBITRARY COMPLEX MATRIX BY USE OF THE QR ALGOR- DALMA008
C        ITHM AND THE WIELANDT INVERSE POWER METHOD FOR VECTORS.       DALMA009
C                                                                      DALMA010
C    USAGE                                                             DALMA011
C        CALL DALMAT (A,VAL,M,IA,NCAL)                                 DALMA012
C                                                                      DALMA013
C    DESCRIPTION OF PARAMETERS                                         DALMA014
C        A   :  INPUT MATRIX, DOUBLY-DIMENSIONED, COMPLEX*16 IN TYPE,  DALMA015
C               ON RETURN FROM DALMAT IT WILL CONTAIN THE COMPLEX EIGEN-DALMA016
C               VECTORS OF THE INPUT MATRIX,                           DALMA017
C        VAL :  SINGLY-DIMENSIONED COMPLEX*16 ARRAY IN WHICH DALMAT    DALMA018
C               RETURNS THE EIGENVALUES OF THE INPUT MATRIX, VAL(I)    DALMA019
C               CORRESPONDS WITH THE EIGENVECTOR A(J,I),J=1,M.         DALMA020
C        M   :  ORDER OF INPUT MATRIX, INTEGER*4.                      DALMA021
C        IA  :  IS THE ROW(FIRST) DIMENSION OF A IN THE CALLING PROGRAMDALMA022
C        NCAL:  INTEGER INDICATOR; UPON RETURN FROM DALMAT IT CONTAINS DALMA023
C               THE NUMBER OF EIGENVALUES WITH CORRESPONDING EIGEN-    DALMA024
C               VECTORS WHICH HAVE BEEN CALCULATED.                    DALMA025
C                                                                      DALMA026
C    REMARKS                                                           DALMA027
```

```
C         EIGENVALUES ARE NOT NECESSARILY CALCULATED IN ANY ABSOLUTE      DALMA029
C         ALGEBRAIC ORDER.                                               DALMA030
C         IN CASE OF NON-CONVERGENCE DALMAT RETURNS NCAL,LT,M. IN THIS   DALMA031
C         CASE USER CAN SCALE THE INPUT MATRIX OR SHIFT VALUES BY ADDING DALMA032
C         A CONSTANT TO THE DIAGONAL OF A.                               DALMA033
C                                                                        DALMA034
C**********************************************************************DALMA035
C                                                                        DALMA043
      N=M                                                                DALMA060
      NCAL=N                                                             DALMA061
      IF(N.NE.1)GO TO 1                                                  DALMA062
      LAMBDA(1)=A(1,1)                                                   DALMA063
      A(1,1)=1.                                                          DALMA064
      GO TO 57                                                           DALMA065
    1 ICOUNT=0                                                           DALMA066
      SHIFT(1)=0.                                                        DALMA067
      IF(N.NE.2)GO TO 4                                                  DALMA068
    2 TEMP=(A(1,1)+A(2,2)+CSQRT((A(1,1)+A(2,2))**2-                      DALMA069
     14.*(A(2,2)*A(1,1)-A(2,1)*A(1,2))))/2.                             DALMA070
      IF(REAL(TEMP).NE.0..OR.AIMAG(TEMP).NE.0.)GO TO 3                   DALMA071
      LAMBDA(M)=SHIFT(1)                                                 DALMA072
      LAMBDA(M-1)=A(1,1)+A(2,2)+SHIFT(1)                                 DALMA073
      GO TO 37                                                           DALMA074
    3 LAMBDA(M)=TEMP+SHIFT(1)                                            DALMA075
      LAMBDA(M-1)=(A(2,2)*A(1,1)-A(2,1)*A(1,2))/(LAMBDA(M)-SHIFT(1))     DALMA076
     1   +SHIFT(1)                                                       DALMA077
      GO TO 37                                                           DALMA078
C                                                                        DALMA079
CC      REDUCE MATRIX A TO HESSENBERG FORM                               DALMA080
C                                                                        DALMA081
    4 NM2=N-2                                                            DALMA082
      DO 15 R=1,NM2                                                      DALMA083
      RP1=R+1                                                            DALMA084
      RP2=R+2                                                            DALMA085
      ABIG=0.                                                            DALMA086
      INT(R)=RP1                                                         DALMA087
      DO 5 I=RP1,N                                                       DALMA088
      ABSSQ=REAL(A(I,R))**2+AIMAG(A(I,R))**2                             DALMA089
      IF(ABSSQ.LE.ABIG)GO TO 5                                           DALMA090
      INT(R)=I                                                           DALMA091
      ABIG=ABSSQ                                                         DALMA092
    5 CONTINUE                                                           DALMA093
      INTER=INT(R)                                                       DALMA094
      IF(INTER.EQ.RP1)GO TO 8                                            DALMA095
      IF(ABIG.EQ.0.)GO TO 15                                             DALMA096
      DO 6 I=R,N                                                         DALMA097
      TEMP=A(RP1,I)                                                      DALMA098
      A(RP1,I)=A(INTER,I)                                                DALMA099
```

```
    6 A(INTER,I)=TEMP                                                       DALMA100
      DO 7 I=1,N                                                            DALMA101
      TEMP=A(I,RP1)                                                         DALMA102
      A(I,RP1)=A(I,INTER)                                                   DALMA103
    7 A(I,INTER)=TEMP                                                       DALMA104
    8 DO 9 I=RP2,N                                                          DALMA105
      MULT(I)=A(I,R)/A(RP1,R)                                               DALMA106
    9 A(I,R)=MULT(I)                                                        DALMA107
      DO 11 I=1,RP1                                                         DALMA108
      TEMP=0.                                                               DALMA109
      DO 10 J=RP2,N                                                         DALMA110
   10 TEMP=TEMP+A(I,J)*MULT(J)                                              DALMA111
   11 A(I,RP1)=A(I,RP1)+TEMP                                                DALMA112
      DO 13 I=RP2,N                                                         DALMA113
      TEMP=0.                                                               DALMA114
      DO 12 J=RP2,N                                                         DALMA115
   12 TEMP=TEMP+A(I,J)*MULT(J)                                              DALMA116
   13 A(I,RP1)=A(I,RP1)+TEMP-MULT(I)*A(RP1,RP1)                             DALMA117
      DO 14 I=RP2,N                                                         DALMA118
      DO 14 J=RP2,N                                                         DALMA119
   14 A(I,J)=A(I,J)-MULT(I)*A(RP1,J)                                        DALMA120
   15 CONTINUE                                                              DALMA121
C                                                                           DALMA122
C     CALCULATE EPSILON                                                     DALMA123
C                                                                           DALMA124
      EPS=0.                                                                DALMA125
      DO 16 I=1,N                                                           DALMA126
   16 EPS=EPS+CABS(A(1,I))                                                  DALMA127
      DO 18 I=2,N                                                           DALMA128
      SUM=0.                                                                DALMA129
      IM1=I-1                                                               DALMA130
      DO 17 J=IM1,N                                                         DALMA131
   17 SUM=SUM+CABS(A(I,J))                                                  DALMA132
   18 IF(SUM.GT.EPS)EPS=SUM                                                 DALMA133
      EPS=SQRT(FLOAT(N))*EPS*1.D-12                                         DALMA134
      IF(EPS.EQ.0.)EPS=1.D-12                                               DALMA135
      DO 19 I=1,N                                                           DALMA136
      DO 19 J=1,N                                                           DALMA137
   19 H(I,J)=A(I,J)                                                         DALMA138
   20 IF(N.NE.1)GO TO 21                                                    DALMA139
      LAMBDA(M)=A(1,1)+SHIFT(1)                                             DALMA140
      GO TO 37                                                              DALMA141
   21 IF(N.EQ.2)GO TO 2                                                     DALMA142
   22 MN1=M-N+1                                                             DALMA143
      IF(REAL(A(N,N)).NE.0..OR.AIMAG(A(N,N)).NE.0.)                         DALMA144
     1 IF(ABS(REAL(A(N,N-1)/A(N,N)))+ABS(AIMAG(A(N,N-1)/A(N,N)))-1.D-9)     DALMA145
     2 24,24,23                                                             DALMA146
   23 IF(ABS(REAL(A(N,N-1)))+ABS(AIMAG(A(N,N-1))).GE.EPS)GO TO 25           DALMA147
```

```
   24 LAMBDA(MN1)=A(N,N)+SHIFT(1)                                                    DALMA148
      ICOUNT=0                                                                      DALMA149
      N=N-1                                                                         DALMA150
      GO TO 21                                                                      DALMA151
C                                                                                   DALMA152
CC    DETERMINE SHIFT                                                               DALMA153
C                                                                                   DALMA154
   25 SHIFT(2)=(A(N-1,N-1)+A(N,N)+CSQRQ((A(N-1,N-1)+A(N,N))**2                      DALMA155
     1 -4.*(A(N,N)*A(N-1,N-1)-A(N,N-1)*A(N-1,N))))/2.                               DALMA156
      IF(REAQ(SHIFT(2)).NE.0..OR.AIMAQ(SHIFT(2)).NE.0.)GO TO 26                     DALMA157
      SHIFT(3)=A(N-1,N-1)+A(N,N)                                                    DALMA158
      GO TO 27                                                                      DALMA159
   26 SHIFT(3)=(A(N,N)*A(N-1,N-1)-A(N,N-1)*A(N-1,N))/SHIFT(2)                       DALMA160
   27 IF(CABQ(SHIFT(2)-A(N,N)).LT.CABQ(SHIFT(3)-A(N,N)))GO TO 28                    DALMA161
      INDEX=3                                                                       DALMA162
      GO TO 29                                                                      DALMA163
   28 INDEX=2                                                                       DALMA164
   29 IF(CABQ(A(N-1,N-2)).GE.EPS)GO TO 30                                           DALMA165
      LAMBDA(MN1)=SHIFT(2)+SHIFT(1)                                                 DALMA166
      LAMBDA(MN1+1)=SHIFT(3)+SHIFT(1)                                               DALMA167
      ICOUNT=0                                                                      DALMA168
      N=N-2                                                                         DALMA169
      GO TO 20                                                                      DALMA170
   30 SHIFT(1) = SHIFT (1) + SHIFT(INDEX)                                           DALMA171
      DO 31 I=1,N                                                                   DALMA172
   31 A(I,I)=A(I,I)-SHIFT(INDEX)                                                    DALMA173
C                                                                                   DALMA174
CC    PERFORM GIVENS ROTATIONS, QR ITERATES                                         DALMA175
C                                                                                   DALMA176
      IF(ICOUNT.LE.10)GO TO 32                                                      DALMA177
      NCAL=M-N                                                                      DALMA178
      GO TO 37                                                                      DALMA179
   32 NM1=N-1                                                                       DALMA180
      TEMP1=A(1,1)                                                                  DALMA181
      TEMP2=A(2,1)                                                                  DALMA182
      DO 36 R=1,NM1                                                                 DALMA183
      RP1=R+1                                                                       DALMA184
      RHO=SQRQ(REAQ(TEMP1)**2+AIMAQ(TEMP1)**2+                                      DALMA185
     1 REAQ(TEMP2)**2+AIMAQ(TEMP2)**2)                                              DALMA186
      IF(RHO.EQ.0.)GO TO 36                                                         DALMA187
      COS=TEMP1/RHO                                                                 DALMA188
      SIN=TEMP2/RHO                                                                 DALMA189
      INDEX=MAXO(R-1,1)                                                             DALMA190
      DO 33 I=INDEX,N                                                               DALMA191
      TEMP=CONJQ(COS)*A(R,I)+CONJQ(SIN)*A(RP1,I)                                    DALMA192
      A(RP1,I)=-SIN*A(R,I)+COS*A(RP1,I)                                             DALMA193
   33 A(R,I)=TEMP                                                                   DALMA194
      TEMP1=A(RP1,RP1)                                                              DALMA195
```

```
      TEMP2=A(R+2,R+1)                                                          DALMA196
      DO 34 I=1,R                                                               DALMA197
      TEMP=COS*A(I,R)+SIN*A(I,RP1)                                              DALMA198
      A(I,RP1)=-CONJG(SIN)*A(I,R)+CONJG(COS)*A(I,RP1)                           DALMA199
   34 A(I,R)=TEMP                                                               DALMA200
      INDEX=MINO(R+2,N)                                                         DALMA201
      DO 35 I=RP1,INDEX                                                         DALMA202
      A(I,R)=SIN*A(I,RP1)                                                       DALMA203
   35 A(I,RP1)=CONJG(COS)*A(I,RP1)                                              DALMA204
   36 CONTINUE                                                                  DALMA205
      ICOUNT=ICOUNT+1                                                           DALMA206
      GO TO 22                                                                  DALMA207
C                                                                               DALMA208
CCC   CALCULATE VECTORS                                                         DALMA209
C                                                                               DALMA210
   37 IF(NCAL.EQ.0)GO TO 57                                                     DALMA211
      N=M                                                                       DALMA212
      NM1=N-1                                                                   DALMA213
      IF(N.NE.2)GO TO 38                                                        DALMA214
      EPS=DMAX1(CABS(LAMBDA(1)),CABS(LAMBDA(2)))*1.D-8                          DALMA215
      IF(EPS.EQ.0.)EPS=1.D-12                                                   DALMA216
      H(1,1)=A(1,1)                                                             DALMA217
      H(1,2)=A(1,2)                                                             DALMA218
      H(2,1)=A(2,1)                                                             DALMA219
      H(2,2)=A(2,2)                                                             DALMA220
   38 DO 55 L=1,NCAL                                                            DALMA221
      DO 40 I=1,N                                                               DALMA222
      DO 39 J=1,N                                                               DALMA223
   39 HL(I,J)=H(I,J)                                                            DALMA224
   40 HL(I,I)=HL(I,I)-LAMBDA(L)                                                 DALMA225
      DO 44 I=1,NM1                                                             DALMA226
      MULT(I)=0.                                                                DALMA227
      INTH(I)=.FALSE.                                                           DALMA228
      IP1=I+1                                                                   DALMA229
      IF(CABS(HL(I+1,I)).LE.CABS(HL(I,I)))GO TO 42                              DALMA230
      INTH(I)=.TRUE.                                                            DALMA231
      DO 41 J=I,N                                                               DALMA232
      TEMP=HL(I+1,J)                                                            DALMA233
      HL(I+1,J)=HL(I,J)                                                         DALMA234
   41 HL(I,J)=TEMP                                                              DALMA235
   42 IF(REAL(HL(I,I)).EQ.0..AND.AIMAG(HL(I,I)).EQ.0.)GO TO 44                  DALMA236
      MULT(I)=-HL(I+1,I)/HL(I,I)                                                DALMA237
      DO 43 J=IP1,N                                                             DALMA238
   43 HL(I+1,J)=HL(I+1,J)+MULT(I)*HL(I,J)                                       DALMA239
   44 CONTINUE                                                                  DALMA240
      DO 45 I=1,N                                                               DALMA241
   45 VECT(I)=1.                                                                DALMA242
      TWICE=.FALSE.                                                             DALMA243
```

```
46 IF(REAQ(HL(N,N)).EQ.0..AND.AIMAQ(HL(N,N)).EQ.0.)HL(N,N)=EPS          DALMA244
   VECT(N)=VECT(N)/HL(N,N)                                             DALMA245
   DO 49 I=1,NM1                                                       DALMA246
   K=N-I                                                               DALMA247
   DO 47 J=K,NM1                                                       DALMA248
47 VECT(K)=VECT(K)-HL(K,J+1)*VECT(J+1)                                 DALMA249
   IF(REAQ(HL(K,K)).EQ.0..AND.AIMAQ(HL(K,K)).EQ.0.)HL(K,K)=EPS         DALMA250
48 VECT(K)=VECT(K)/HL(K,K)                                             DALMA251
   BIG=0.                                                              DALMA252
   DO 49 I=1,N                                                         DALMA253
   SUM=ABQ(REAQ(VECT(I)))+ABQ(AIMAQ(VECT(I)))                          DALMA254
49 IF(SUM.GT.BIG)BIG=SUM                                               DALMA255
   DO 50 I=1,N                                                         DALMA256
50 VECT(I)=VECT(I)/BIG                                                 DALMA257
   IF(TWICE)GO TO 52                                                   DALMA258
   DO 51 I=1,NM1                                                       DALMA259
   IF(.NOT.INTH(I))GO TO 51                                            DALMA260
   TEMP=VECT(I)                                                        DALMA261
   VECT(I)=VECT(I+1)                                                   DALMA262
   VECT(I+1)=TEMP                                                      DALMA263
51 VECT(I+1)=VECT(I+1)+MULT(I)*VECT(I)                                 DALMA264
   TWICE=.TRUE.                                                        DALMA265
   GO TO 46                                                            DALMA266
52 IF(N.EQ.2)GO TO 55                                                  DALMA267
   NM2=N-2                                                             DALMA268
   DO 54 I=1,NM2                                                       DALMA269
   N1I=N-1-I                                                           DALMA270
   NI1=N-I+1                                                           DALMA271
   DO 53 J=NI1,N                                                       DALMA272
53 VECT(J)=H(J,N1I)*VECT(N1I+1)+VECT(J)                                DALMA273
   INDEX=INT(N1I)                                                      DALMA274
   TEMP=VECT(N1I+1)                                                    DALMA275
   VECT(N1I+1)=VECT(INDEX)                                             DALMA276
54 VECT(INDEX)=TEMP                                                    DALMA277
55 DO 56 I=1,N                                                         DALMA278
56 A(I,L)=VECT(I)                                                      DALMA279
57 RETURN                                                              DALMA280
   END                                                                 DALMA281
   REAL FUNCTION CABQ*8 (Z)                                            DALMA282
   COMPLEX*16 Z                                                        DALMA283
   CABQ = CDABS(Z)                                                     DALMA284
   RETURN                                                              DALMA285
   END                                                                 DALMA286
   REAL FUNCTION SQRQ*8 (X)                                            DALMA287
   REAL * 8 X                                                          DALMA288
   SQRQ = DSQRT(X)                                                     DALMA289
   RETURN                                                              DALMA290
   END                                                                 DALMA291
```

```
      REAL FUNCTION FLOAQ*8(I)                                          DALMA292
      FLOAQ = DFLOAT (I)                                                DALMA293
      RETURN                                                            DALMA294
      END                                                               DALMA295
      REAL FUNCTION ABQ*8(X)                                            DALMA296
      REAL*8 X                                                          DALMA297
      ABQ = DABS(X)                                                     DALMA298
      RETURN                                                            DALMA299
      END                                                               DALMA300
      COMPLEX FUNCTION CSQRQ*16 (Z)                                     DALMA301
      COMPLEX*16 Z                                                      DALMA302
      CSQRQ = CDSQRT(Z)                                                 DALMA303
      RETURN                                                            DALMA304
      END                                                               DALMA305
      COMPLEX FUNCTION CONJQ*16 (Z)                                     DALMA306
      COMPLEX*16 Z                                                      DALMA307
      CONJQ = DCONJG (Z)                                                DALMA308
      RETURN                                                            DALMA309
      END                                                               DALMA310
      REAL FUNCTION REAQ*8 (Z)                                          DALMA311
      COMPLEX *16 Z                                                     DALMA312
      REAQ = Z                                                          DALMA313
      RETURN                                                            DALMA314
      END                                                               DALMA315
      REAL FUNCTION AIMAQ*8 (Z)                                         DALMA316
      COMPLEX *16 Z                                                     DALMA317
      AIMAQ = Z * (0.D0,-1.D0)                                          DALMA318
      RETURN                                                            DALMA319
      END                                                               DALMA320
```

BIBLIOGRAPHY

Arakawa, A., "Computational Design for Long-Term Numerical Integration of the Equations of Fluid Motion: Two-Diemsnional Incompressible Flow. Part 1.," Jour. Comp. Physics, 1, 1, pp. 119-143, (1966).

Aziz, K. and Hellums, J. D., "Numerical Solution of the Three-Dimensional Equations of Motion for Natural Convection," Phys. Fluids, 10, 7, pp. 314-324, (1967).

Cooley, J. W. and Tukey, J. W., "An Algorithm for the Machine Calculation of Complex Fourier Series," Math. Comp., 19, 90, pp. 297-301, (1965).

Davey, A. and Drazin, P. G., "The Stability of Poiseuille Flow in a Pipe," Jour. Fluid Mech., 36, 2, pp. 209-218, (1969).

Deardorff, J. W., "A Numerical Study of Three-dimensional Turbulent Channel Flow at Large Reynolds Numbers," Jour. Fluid Mech., 41, 2, pp. 453-480, (1970).

Dixon, T. N., "A Study on Stability and Incipient Turbulence in Poiseuille and Plane Poiseuille Flow by Finite Difference Simulation," Ph.D. Dissertation, Rice University, (1966).

Douglas, J. and Rachford, N. H., "On the Numerical Solution of Heat Conduction Problems in Two and Three Space Variables," Trans. American Math. Soc., 82, pp. 421-439, (1956).

Du Fort, E. C. and Frankel, S. P., "Stability Conditions in the Numerical Treatment of Parabolic Differential Equations," Math. Tables and Other Aids to Computation, 7, pp. 135-152, (1953).

Francis, J. G. F., "The Q R Transformation, Parts I and II," Computer Jour., 4, pp. 265-271, 332-345, (1962).

Fromm, J. E., A Method for Computing Non-Steady, Incompressible, Viscous Fluid Flows, Los Alamos Scientific Laboratory Report No. LA-2910, (1963).

Fischer, G., "A Survey of Finite-Difference Approximations to the Primitive Equations," Monthly Weather Rev., 93, 1, pp. 1-9, (1965).

Galloway, F. M., "On a Finite-Difference Approach to Turbulence Problems," Ph.D. Dissertation, Case Western Reserve Univ., (1968).

Gawain, T. H., Pritchett, J. W., "A Unified Heuristic Model of Fluid Turbulence," to be published Jour. Comp. Phys., (1970).

Grosch, C. E., and Salwen, H., "The Stability of Steady and Time-dependent Plane Poiseuille Flow," Jour. Fluid Mech., 34, 1, pp. 177-205, (1968).

Hamming, R. W., Numerical Methods for Scientists and Engineers, McGraw-Hill, (1962).

Harlow, F. H. and Welch, J. E., "Numerical Calculation of Time Dependent Viscous Incompressible Flow of Fluid with Free Surface," Phys. Fluids, 8, 12, pp. 2182-2189, (1964).

Harlow, F. H. and Welch, J. E., The MAC Method, Los Alamos Scientific Laboratory Report No. LA-3425, (1965).

Heisenberg, W., "Über Stabilitat und Turbulenz von Flussigkeitsströmen," Ann. Phys., 4, 74, pp. 577-627, (1924).

Kraichnan, R. H., "Inertial Ranges in Two-Dimensional Turbueence," Phys. Fluids, pp. 10, 7, (1967).

Kurihara, Y., "On the use of Implicit and Iterative Methods for the Time Integration of the Wave Equation," Monthly Weather Rev., 93, 1, pp. 33-46 (1965).

Lilly, D. K., "On the Computational Stability of Numerical Solutions of Time-Dependent Non-linear Geophysical Fluid Dynamics Problems," Monthly Weather Rev., 93, 1, pp. 11-26, (1965).

Lilly, D. K., Numerical Solution of Field Problems in Continuum Physics, VII SIAM-AMS Proceedings, American Mathematical Society, pp. 41-53, (1970).

Lin, C. C., "On the Stability of Two-dimensional Parallel Flows," Parts I, II, III, Quarterly Applied Math., 3, pp. 117-142, pp. 218-234, pp. 277-301, (1945).

Lin, C. C., Theory of Hydrodynamic Stability, Cambridge Univ. Press, (1955).

Meksyn, D. and Stuart, J. T., "Stability of Viscous Motion between Parallel Planes for Finite Disturbances," Proc. Royal Soc., A 208, pp. 517-526, (1951).

Nachtshein, P. R., An Initial Value Method for the Numerical Treatment of the Orr-Sommerfeld Equation for the Case of Plane Poiseuille Flow, NASA TN D-2414, (1964).

O'Brien, G. G., Hyman, M. A. and Kaplan, S., "A Study of the Numerical Solutions of Partial Differential Equations," Jour. Math. and Phys., 29, 4, pp. 223-251, (1951).

Pai, S. I., Viscous Flow Theory II, Turbulent Flow, D. Van Nostrand Co., (1957).

Parlett, B. N., Mathematical Methods for Digital Computers, Vol. II, Ed. A. Ralston and H. Wilf, Wiley & Sons Inc., (1967).

Peaceman, D. W. and Rachford, H. H., "The Numerical Solution of Parabolic and Elliptic Differential Equations," Jour. S. I. A. M., 3, pp. 28-35, (1955).

Pearson, C. E., "A Computational Method for Viscous Flow Problems," Jour. Fluid Mech., 21, 4, pp. 611-622, (1965).

Pekeris, C. L. and Shkoller, "Stability of Plane Poiseuille Flow to Periodic Disturbances of Finite Amplitude," Jour. Fluid Mech., 39, 3, pp. 611-627, (1969).

Pekeris, C. L. and Shkoller, "The Neutral Curves for Periodic Perturbations of Finite Amplitude of Plane Poiseuille Flow," Jour. Fluid Mech., 39, 3, pp. 629-639, (1969).

Pekeris, C. L. and Shkoller, "Stability of Plane Poiseuille Flow to Periodic Disturbances of Finite Amplitude in the Vicinity of the Neutral Curve," Jour. Fluid Mech., 29, 1, pp. 31-38, (1967).

Potter, M. C., "Stability of Plane Couette-Poiseuille Flow," Jour. Fluid Mech., 24, 3, pp. 609-619, (1966).

Phillips, N. A., The Atmosphere and the Sea in Motion, pp. 501-504, Rockefeller Inst. Press, (1959).

Prandtl, L., "Über die Ausgebildete Turbulenz," Z. angew Math. u Mech., 5, 136, (1925).

Reynolds, O., "An Experimental Investigation of the Circumstances which Determine whether the Motion of Water shall be Direct or Sinous, and the Law of Resistance in Parallel Channels," Phil. Trans. Royal Soc., 174, pp. 935-982, (1883).

Reynolds, W. C., and Potter, M. C., "Finite Amplitude Instability of Parallel Shear Flows," Jour. Fluid Mech., 27, 3, pp. 465-492, (1967).

Richtmyer, R. D., and Morton, K. W., Difference Methods for Initial-Value Problems, 2nd Ed., Interscience, (1967).

Schensted, I. V., "Contributions to the Theory of Hydrodynamic Stability," Ph.D. Dissertation, Univ. Michigan, (1960).

Smith, G. D., Numerical Solution of Partial Differential Equations, Oxford Univ. Press, (1965).

Stuart, J. T., "On the Non-linear Mechanics of Hydrodynamic Stability," Jour. Fluid Mech., 4, 1, pp. 1-21, (1958).

Stuart, J. T., "On the Non-linear Mechanics of Wave Disturbances in Stable and Unstable Parallel Flows," Part 1. "The basic Behavior in Plane Poiseuille Flow," Jour. Fluid Mech., 9, 2, pp. 353-370, (1960).

Thomas, L. H., "The Stability of Plane Poiseuille Flow," Phys. Rev., 91, 4, pp. 780, (1953).

Varga, R. S., Matrix Iterative Analysis, Prentice-Hall, (1962).

Watson, J., "On the Non-linear Mechanics of Wave Disturbances in Stable and Unstable parallel Flows, Part 2. The Development of a Solution for Plane Poiseuille Flow and for Plane Couette Flow," Jour. Fluid Mech., 9, 2, pp. 371-389, (1960).

Wilkinson, J. A., The Algebraic Eigenvalue Problem, pp. 485-569, Claredon Press, (1965).

Williams, C. P., "Numerical Integration of the Three-dimensional Navier-Stokes Equations for Incompressible Flow," Jour. Fluid Mech., 37, 4, pp. 727-750, (1969).

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Chairman, Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
4.	Professor T. H. Gawain, Code 57Gn Department of Aeronautics Naval Postgraduate School Monterey, California 93940	5
5.	Professor D. J. Collins, Code 57Co Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
6.	Professor F. D. Faulkner, Code 53Fa Department of Mathematics Naval Postgraduate School Monterey, California 93940	1
7.	Associate Professor J. A. Miller, Code 57Mo Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
8.	Associate Professor J. V. Sanders, Code 61Sd Department of Physics Naval Postgraduate School Monterey, California 93940	1
9.	Associate Professor R. T. Williams, Code 51Wu Department of Meteorology Naval Postgraduate School Monterey, California 93940	1
10.	Lieutenant Commander George D. O'Brien, USN 4817 Bayard Boulevard Washington, D. C. 20016	3

11. Deputy Director — 1
Advanced Research Projects Agency
Attn: Dr. S. Lukasik

12. Mr. Ronald Bailey — 1
Theoretical Branch
NASA Ames Research Center
Moffett Field, California 94035

13. Prof. W. H. Clark — 1
Department of Aeronautics
Naval Postgraduate School
Monterey, California 93940

14. Dr. F. M. Galloway — 1
Case Western Reserve University
Cleveland, Ohio 44106

15. Information Research Associates — 2
2140 Shattuck Avenue
Berkeley, California 94720

16. Los Alamos Scientific Laboratory — 1
Los Alamos, New Mexico 87544
Attn: Dr. F. Harlow

17. Commander — 2
Naval Air Systems Command
Attn: Dr. E. S. Lamar - 1
Dr. H. J. Mueller - 1
Navy Department
Washington, D. C. 20360

18. Commanding Officer — 2
Naval Ship Research and Development Center
Carderoc, Maryland
Attn: Dr. F. Frenkiel - 1
Dr. J. Schott - 1

19. Office of Naval Research — 4
Navy Department
Washington, D. C. 20360
Attn: Physical Sciences Div., E. H. Weinberg, Code 420 - 1
Mathematical & Information Sciences Div., R. J. Lindegard, Code 430 - 1
Morton Cooper, Code 438 - 1
Ralph Cooper - 1

20. Dr. W. C. Reynolds — 1
Dept. of Mechanical Engineering
Standord University
Stanford, California 94034

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Naval Postgraduate School Monterey, California 93940	Unclassified
	2b. GROUP

3. REPORT TITLE

A Numerical Investigation of Finite-Amplitude Disturbances in a Plane Poiseuille Flow

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)

Doctoral Dissertation, June 1970

5. AUTHOR(S) (First name, middle initial, last name)

George Donoghue O'Brien

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1970	204	49

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

A consistent and stable finite-difference approximation of the vorticity transport equations has been applied to a plane Poiseuille flow with spatially periodic disturbances of finite amplitude.

By application of the discrete Fourier transform to the solution of a Poisson equation significant improvements in the speed and accuracy of the calculations have been obtained. Numerical tests indicate that the usual iterative schemes are inadequate if a large number of mesh points areemployed.

The effect of mesh size has been determined by extensive calculations based on a consistent second-order representation of the linear eigenvalue problem. The results indicate that a fine mesh is required to accurately represent the behavior of even the large-scale unstable motions. The stability and accuracy of the present formulation was demonstrated, in the linear range, by comparison with the established results of the linear theory.

Calculations in the non-linear range, at a moderate value of the Reynolds number, indicate the existence of an equilibrium state. The calculated values of the mean spectral density of kinetic energy are consistent with the exponential law predicted by Kraichnan for two-dimensional turbulence. The program, developed in this investigation, provides a means, of demonstrated accurary, for the extensive investigation of two dimensional turbulence in a shear flow.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

A-31408

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Hydrodynamic stability						
Finite-differences						
Turbulence						
Plane Poiseuille flow						

19435

Thesis 124643
019 O'Brien
c.1 A numerical investiga-
tion of finite-ampli-
tude disturbances in
a plane Poiseuille flow.

19435

ga-

124643

Thesis
019 O'Brien
c.1 A numerical investiga-
tion of finite-ampli-
tude disturbances in
a plane Poiseuille flow.

thesO19
A numerical investigation of finite-ampl
3 2768 001 97610 3
DUDLEY KNOX LIBRARY